

AD-A096 092

BOLT BERANEK AND NEWMAN INC CAMBRIDGE MA
DESIGN AND REAL-TIME IMPLEMENTATION OF A ROBUST
DEC 80 J J WOLF, K D FIELD, W H RUSSELL

F/G 17/2.1
APC CODER FOR S--ETC(U)
DCA100-79-C-0037

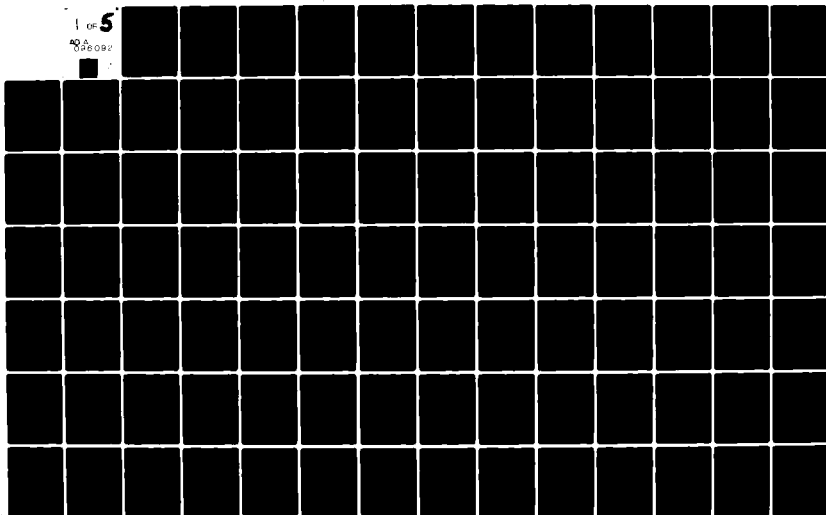
UNCLASSIFIED

BBN-4565-VOL-2

NL

1 OF 5

000000



Bolt Beranek and Newman Inc.



AD A096092

LEVEL

12

Report No. 4565

Design and Real-Time Implementation of a Robust APC Coder
for Speech Transmission Over 16 Kb/s Noisy Channels.

Volume II: Real-Time Implementation.

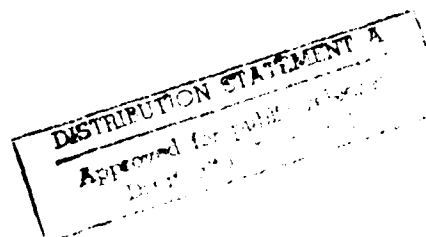
Final Report.

DTIC

MAR 9 1981

December 1980

Prepared for:
Defense Communications Agency



DOC FILE COPY

81 2 00 075

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER BBN Report No. 4565, Vol. II	2. GOVT ACCESSION NO. AD-A696 092	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) DESIGN AND REAL-TIME IMPLEMENTATION OF A ROBUST APC CODER FOR SPEECH TRANSMISSION OVER 16 Kb/s NOISY CHANNELS Vol. II: Real-Time Implementation		5. TYPE OF REPORT & PERIOD COVERED Final Report July 1979 - December 1980
7. AUTHOR(s) J.J. Wolf, K.D. Field, W.H. Russell		6. PERFORMING ORG. REPORT NUMBER BBN Report No. 4565
9. PERFORMING ORGANIZATION NAME AND ADDRESS Bolt Beranek and Newman Inc. 50 Moulton Street Cambridge, MA 02238		8. CONTRACT OR GRANT NUMBER(s) DCA100-79-C-0037 ✓
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Communications Agency Contract Management Division, Code 260 Washington, D.C. 20305		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE December 1980
		13. NUMBER OF PAGES 417
		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Distribution of this document is unlimited. It may be released to the Clearinghouse, Department of Defense, for sale to the general public.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Speech coding, 16 kb/s speech transmission, adaptive predictive coding, digital voice terminal, real-time speech coder, medium-bandwidth speech transmission, array processor		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This report describes the design and development of a real-time robust APC speech coder that transmits high-quality speech over a 16 kb/s channel with bit-error rates of up to 1%. This volume (Vol. II) describes the implementation of the optimized adaptive predictive coding algorithm as a full-duplex real-time speech coder on the CSP Inc. MAP-300 array processing computer. The report documents in detail the MAP-300 program, and it includes complete listings of the real-time coder program.		

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)



SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Report No. 4565

DESIGN AND REAL-TIME IMPLEMENTATION OF A ROBUST APC CODER
FOR SPEECH TRANSMISSION OVER 16 Kb/s NOISY CHANNELS

Final Report

Volume II: Real-Time Implementation

Authors: J. Wolf, K. Field, W. Russell

December 1980

Transmission For	
GR&I	☒
IAB	☐
Unannounced	☐
Notification	☐
by	
Distribution/	
Availability Codes	
Dist	Special
A	

Prepared for:
Defense Communications Agency

TABLE OF CONTENTS

	Page
1. INTRODUCTION	1
2. REAL-TIME IMPLEMENTATION	3
2.1 OVERVIEW	3
2.1.1 System Function	3
2.1.2 System Components	3
2.1.3 System Design	4
2.2 SYSTEM OPERATION	9
2.2.1 Transmitter	9
2.2.1.1 ADAM Scroll Program (ADPROG)	12
2.2.1.2 A/D Interrupt Service (ADAMINT)	15
2.2.1.3 ANALYZE Module	17
2.2.1.4 TMODEM Interrupt Service (TMODEMINT)	27
2.2.1.5 TMODEM and RMODEM I/O Scroll Program (RTPROG)	29
2.2.1.6 Transmitter Input/Output Buffer Initial Sequence	32
2.2.2 Receiver	33
2.2.2.1 RMODEM Scroll Program	36
2.2.2.2 RMODEM Interrupt Service (RMODEMINT)	36
2.2.2.3 SYNTHESIZE Module	45
2.2.2.4 D/A Interrupt Service Routine (AOMINT)	50
2.2.2.5 D/A Scroll Program (DAPROG)	50
2.2.2.6 Receiver Input/Output Buffer Initial Sequence	52
2.3 SYSTEM HARDWARE	54
2.3.1 MAP-300 Hardware	55
2.3.2 Audio and Modem Interface Hardware	55
2.4 SYSTEM SOFTWARE	56
2.4.1 MAP-300 Software Components	57
2.4.1.1 CSPI-Supplied MAP-300 Software	57
2.4.1.2 BBN-written MAP-300 Software	59
2.4.2 PDP-11 Software Components	60
2.4.2.1 CSPI-Supplied PDP-11 Software	61
2.4.2.2 BBN-written PDP-11 Software	61

2.5	SYSTEM TIMING PERFORMANCE	67
3.	SOFTWARE OPERATING PROCEDURES	71
3.1	SOFTWARE EXECUTION PROCEDURE	71
3.1.1	Optional Software Execution Procedure	73
3.1.2	Analog Input Level Setting	74
3.2	SOFTWARE INSTALLATION PROCEDURE	75
4.	REFERENCES	79
APPENDIX A.	BBN-WRITTEN FUNCTION DESCRIPTIONS	81
APPENDIX B.	MAP-300 BUFFERS	110
APPENDIX C.	MAP-300 SCALARS	126
APPENDIX D.	PROGRAM LISTINGS	138

LIST OF FIGURES

FIG. 1.	SYSTEM COMPONENTS	5
FIG. 2.	SYSTEM STRUCTURE	6
FIG. 3.	BACKGROUND PROCESS LOOP	8
FIG. 4.	TRANSMITTER PROCESS	10
FIG. 5.	ADPROG: ADAM SCROLL PROGRAM	14
FIG. 6.	A/D INTERRUPT SERVICE ROUTINE	16
FIG. 7.	TMODEM INTERRUPT SERVICE ROUTINE (TMODEMINT)	30
FIG. 8.	TMODEM AND RMODEM SCROLL PROGRAM FLOW CHART	31
FIG. 9.	RECEIVER PROCESS	34
FIG. 10.	RMODEM INTERRUPT SERVICE MODULE (RMODEMINT)	38
FIG. 11.	FRAME SYNCHRONIZATION INITIALIZATION MODULE (SYNCINIT)	41
FIG. 12.	FRAME SYNCHRONIZATION SEARCH MODULE (SYNCSRCH)	42
FIG. 13.	FRAME SYNCHRONIZATION UPDATE MODULE (SYNCUPDATE)	44
FIG. 14.	D/A INTERRUPT SERVICE ROUTINE (AOMINT)	51
FIG. 15.	DAPROG: AOM SCROLL PROGRAM	53
FIG. 16.	MAP-300 MEMORY ORGANIZATION	58
FIG. 17.	MAP-300 TIMING	69

Bolt Beranek and Newman Inc.

Report No. 4565

LIST OF TABLES

TABLE 1.	TSINK BUFFER FORMAT	27
TABLE 2.	TBITS BUFFER FORMAT	28

1. INTRODUCTION

The purpose of this project was the design and development of a real-time speech coding system that transmits high-quality speech at a data rate of 16 kb/s (kilobits/second). The final report of this project is organized in two volumes. Volume I describes our work in the development and optimization of the speech coding algorithm. Volume II (this volume) describes the implementation of the final optimized speech coding algorithm as a real-time full duplex system on a CSP Inc. (CSPI) MAP-300 signal processing computer and associated hardware.

In the body of this volume we present detailed documentation of the MAP-300 real-time implementation of the speech coding algorithm (Section 2) and instructions on the installation and use of the speech coder software (Section 3). Contained in appendices are function descriptions of BBN-written MAP-300 modules (Appendix A), documentation of the MAP-300 buffers and scalars (Appendices B and C), and listings of the MAP-300 and PDP-11 (FORTRAN) programs that constitute the speech coder system (Appendix D).

Bolt Beranek and Newman Inc.

Report No. 4565

2. REAL-TIME IMPLEMENTATION

2.1 OVERVIEW

This section describes the function, components, and design of the BBN16 real-time speech coder implementation. The later sections in this chapter describe in more detail the operation of the speech coder system, the hardware and software used to construct it, and the real-time performance of the completed system.

2.1.1 System Function

The speech coder system is a full duplex terminal of a complete communication system. It is intended to be connected via a 16 kb/s digital I/O link through a communication channel to an identical system. The speech coder system functions simultaneously as both a transmitter and a receiver.

2.1.2 System Components

The speech coder system contains both hardware and software elements. The hardware elements include a CSP Inc. MAP-300 array processor attached to a PDP-11, a handset including microphone and earphone, a hookswitch, amplifiers and low-pass filters, and digital line interfaces. The software elements include MAP-300 programs, which comprise the real-time software, and the program that runs on the PDP-11, which is used only to load and

initialize the MAP software. Fig. 1 is a block diagram of the system.

2.1.3 System Design

The real-time speech coder consists of six separate foreground processes. The Transmitter requires an analog-in process, an analysis process, and a digital-out process. The Receiver requires a digital-in process, a synthesis process, and an analog-out process. Since all of these processes make use of the MAP-300 CSPU to some extent, a mechanism for scheduling them is necessary. Part of this mechanism is contained in the MAP hardware interrupt structures and the SNAP-II executive program. The rest is implemented using flags in conjunction with a background process running in the CSPU. Fig. 2 is a diagram of these processes. The processes share data buffers and communicate the status of these buffers through flags. Since the four I/O processes are continuous, pairs of double buffers are used to enable the necessary sharing.

Each I/O process includes an interrupt service routine, which handles the flags and transfers data to or from the shared buffers. Two levels of double buffering are provided within each I/O process to maintain acceptable system performance under real-time exception conditions.

The background process is shown in Fig. 3. The ANALYZE and

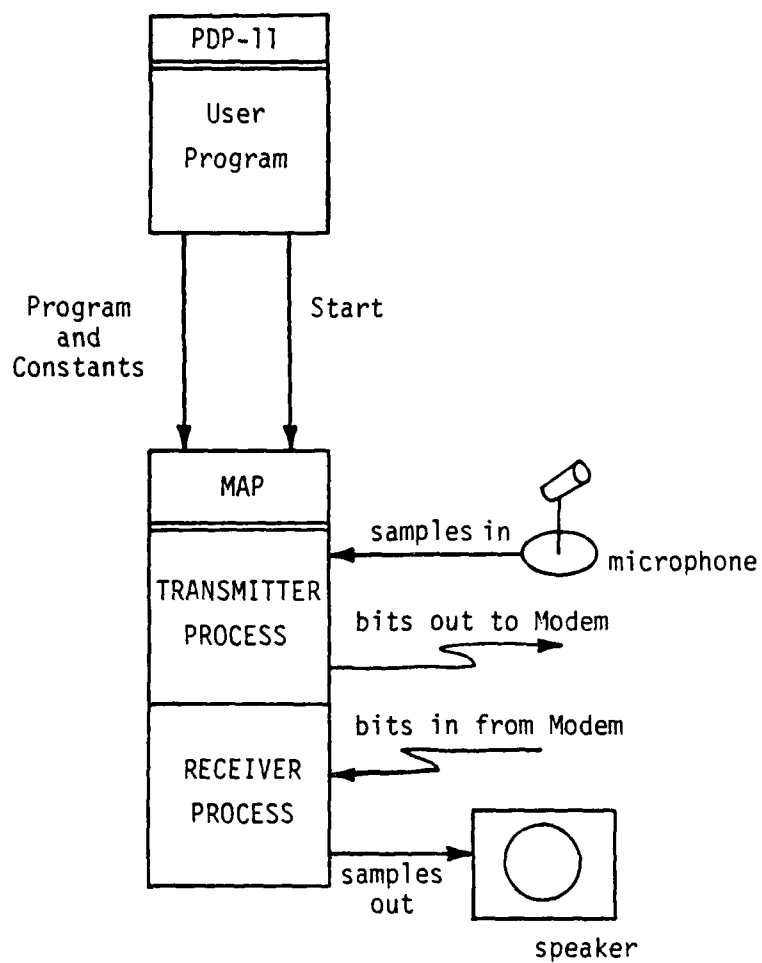


FIG. 1. SYSTEM COMPONENTS

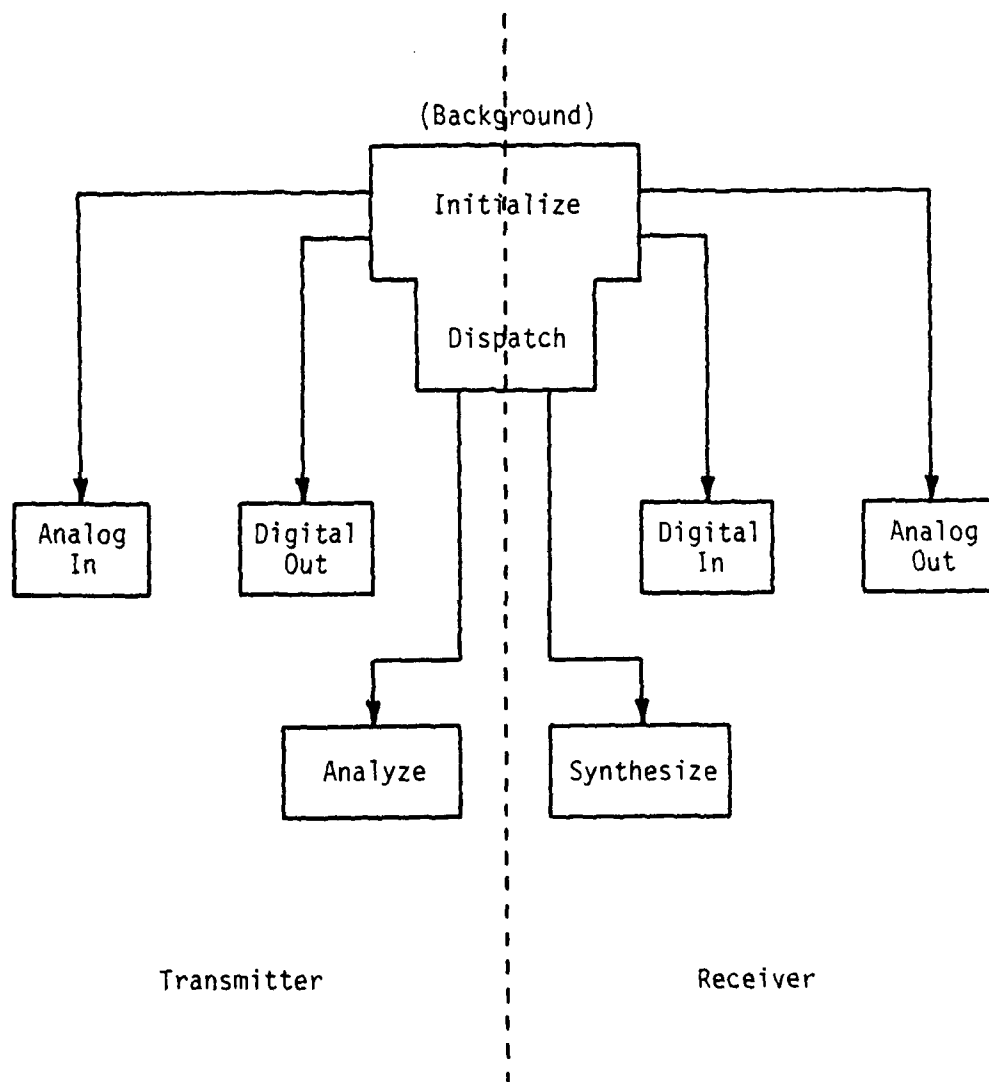


FIG. 2. SYSTEM STRUCTURE

SYNTHESIZE modules are logically asynchronous; they respectively belong to the independent Transmitter and Receiver. However, the modules must be controlled by a sequential machine, the CSPU. The control strategy for allowing each module as much flexibility as possible results in a structure that executes a module only if that module's input and output buffers are available.

The Initialization module is the first module executed, and it is executed only once. It sets all buffer flags to indicate empty, loads and starts the programs in the various I/O Scroll processors, and enables interrupts from these processors.

Control then passes to the basic loop of the background process. This loop executes the ANALYZE and SYNTHESIZE modules when the required I/O buffers are available. For example, the SYNTHESIZEA module will be executed only when the RBITSB buffer is full and the RSINKA buffer is empty.

The ANALYZEA and ANALYZEB modules are functionally equivalent, differing only in their input and output buffers. The structure of the SNAP-II programming environment (specifically, the use of prebound functions for run-time efficiency) does not allow changing the buffers used in a particular function. Therefore, two separate modules have been created to deal with the two sets of TSOURCE and TBITS buffers, A and B. Similarly, two separate SYNTHESIZE modules are required.

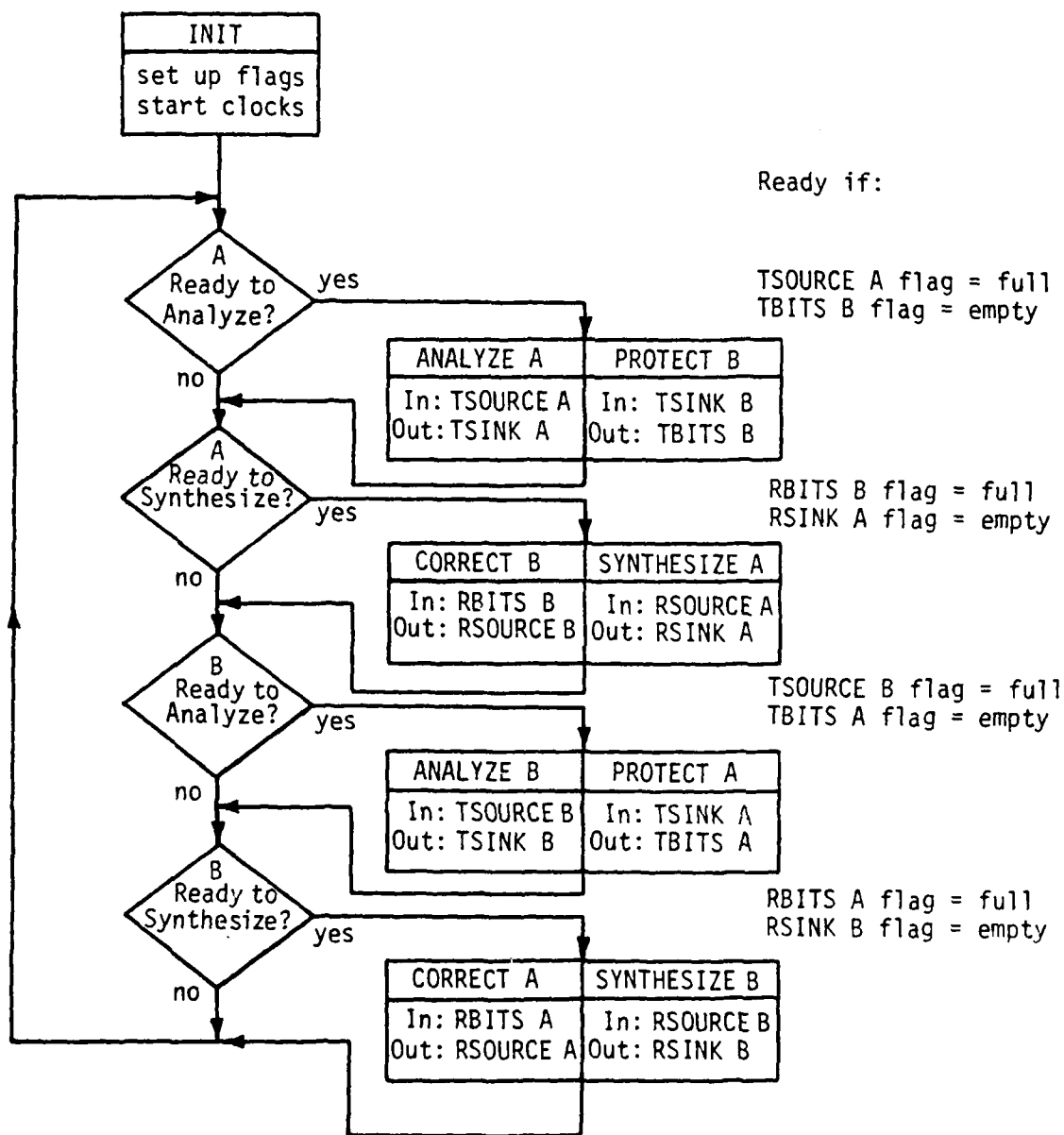


FIG. 3. BACKGROUND PROCESS LOOP

All of the foreground processes are loosely coupled. Execution of neither the ANALYZE nor the SYNTHESIZE module is connected rigidly to any I/O process.

2.2 SYSTEM OPERATION

The speech coder system functions as one terminal of a full duplex digital voice connection. It accepts voice input, digitizes it, and processes it. The processed speech is transmitted as a sequence of bits to a similar terminal. The system also accepts a sequence of bits representing speech transmitted from a remote terminal and processes this sequence to obtain synthetic voice output.

2.2.1 Transmitter

The Transmitter is shown in Fig. 4. The low-pass filtered input speech is fed into an A/D converter contained in an I/O Scroll processor (the ADAM, or Analog Data Acquisition Module), a component processor of the MAP. The program running in this scroll controls the sampling of the speech data, transferring 216 samples (one frame) into each of two buffers, alternately. The sampling rate is 6.621 kHz, so each frame is about 32.625 ms long.

When the ADAM fills one of these buffers, it generates an interrupt to the main MAP processor, the CSPU. The A/D Interrupt

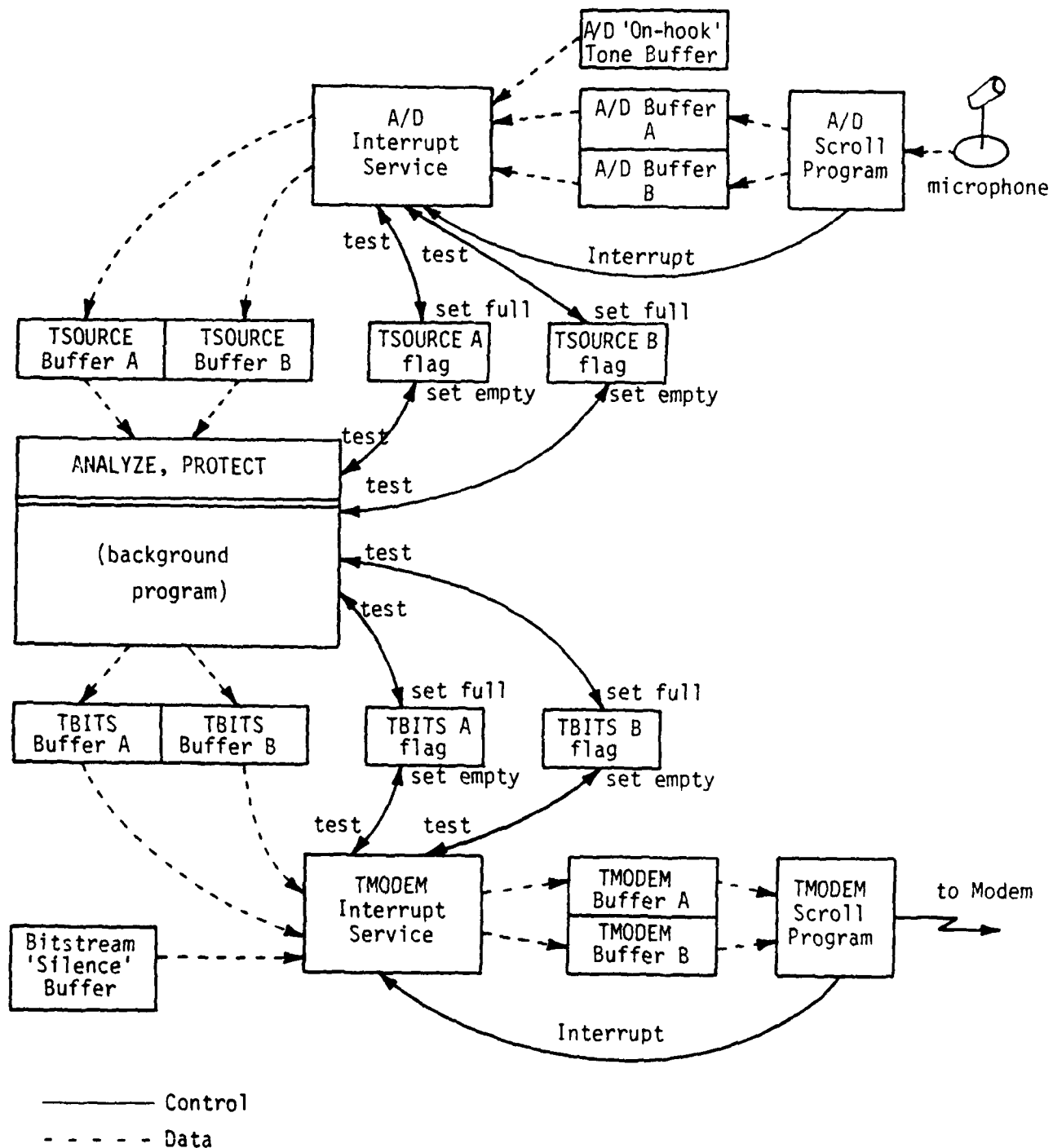


FIG. 4. TRANSMITTER PROCESS

Service routine, which is activated by this interrupt, transfers the data from the just-filled A/D buffer to the current one of the two TSOURCE buffers (if the current one is empty) and sets the corresponding TSOURCEA or TSOURCEB flag to indicate that the buffer is now full. If the current TSOURCE Buffer is not empty, the A/D buffer contents are discarded.

The data in the TSOURCE buffers is used as input by the ANALYZE program module. The Background Process (Section 2.1.3) tests the TSOURCE flags to determine when data is available to the ANALYZE module. After processing a TSOURCE buffer, ANALYZE clears the corresponding flag to indicate that the buffer is empty. The ANALYZE module processes the data, producing coded speech, which is then written into one of the (empty) TSINK buffers. The ANALYZE module runs in the CSPU and AP at background level and is therefore asynchronous with the interrupt service routines.

After ANALYZE fills a TSINK buffer, the PROPAR and PRORES modules transform the data from this buffer into an error-protected bitstream. This bitstream is stored in one of the (empty) TBITS buffers, and the corresponding TBITS flag is set to indicate full. The PROPAR/PRORES operation logically follows ANALYZE, in that data must be processed by ANALYZE before it is available to PROPAR/PRORES. However, the PROPAR and PRORES modules are actually executed concurrently with the next

execution of the ANALYZE module. In fact, they appear as part of the ANALYZE function list. The first time the ANALYZE and PROPAR/PRORES modules are executed, PROPAR/PRORES will operate on a TSINK buffer before it has been filled by ANALYZE; therefore, the TSINK buffers are initialized to coded silence.

The data processed by the PROPAR/PRORES modules will eventually be output to the modem by the TMODEM program, running in another I/O Scroll processor. This program takes data, in the form of a bitstream, from the TMODEM buffers and puts it out to the modem. When the TMODEM buffer is emptied, the TMODEM scroll program generates an interrupt to the CSPU and begins taking data from the other TMODEM buffer.

This interrupt causes execution of the TMODEM Interrupt Service routine. This routine transfers data from a full TBITS buffer to the just-emptied TMODEM buffer and clears the corresponding TBITS flag to indicate empty. If the TBITS buffer is not full, a buffer corresponding to coded silence is transferred to the TMODEM buffer.

2.2.1.1 ADAM Scroll Program (ADPROG)

Speech input is performed by the ADAM, an IOS-2 scroll processor that contains an analog multiplexer, sample/hold, and A/D converter. The ADAM receives both the input speech signal and the input sampling clock from the Speech Processor Interface (SPI). The signal is in the range -5 to +5 volts, and the sampling rate is 6.621 kHz.

The ADAM program (ADPROG) is shown in Fig. 5. The speech samples are double-buffered in A/D input buffers named TADBA and TADBB. (All MAP buffers and most scalars are prefixed with "T" or "R" to indicate that they are used in the speech coder Transmitter or Receiver respectively.) The A/D input buffers are 216 samples long, which corresponds to a frame length of about 32.625 ms. The A/D samples are written into memory in short (16-bit) floating point format. (The ADAM and AOM are capable of working only in 16-bit floating or 16-bit fixed point formats.)

ADPROG sets the ADAM multiplexer to Channel 1 and sets the F1 flag to initiate sampling. ADPROG maintains a pointer into the A/D input buffer, which is initialized to TADBA-1. When a sample is converted, this address pointer is incremented and used to transfer the A/D sample to MAP memory. The address pointer is compared to the buffer end address; when the end has been reached, a line 1 interrupt is passed to the CSPU to signal the filling of the buffer, and the SYNCSTOP register is checked. SYNCSTOP is a register that is set by the CSPU to signal the ADAM to stop sampling. In normal operation, it remains clear, so ADPROG resets the address pointer to the other A/D input buffer (TADBB), and speech sample input proceeds without interruption. Interrupt 1 from the ADAM is used to signal the filling of each successive A/D buffer and the switch to the other A/D buffer.

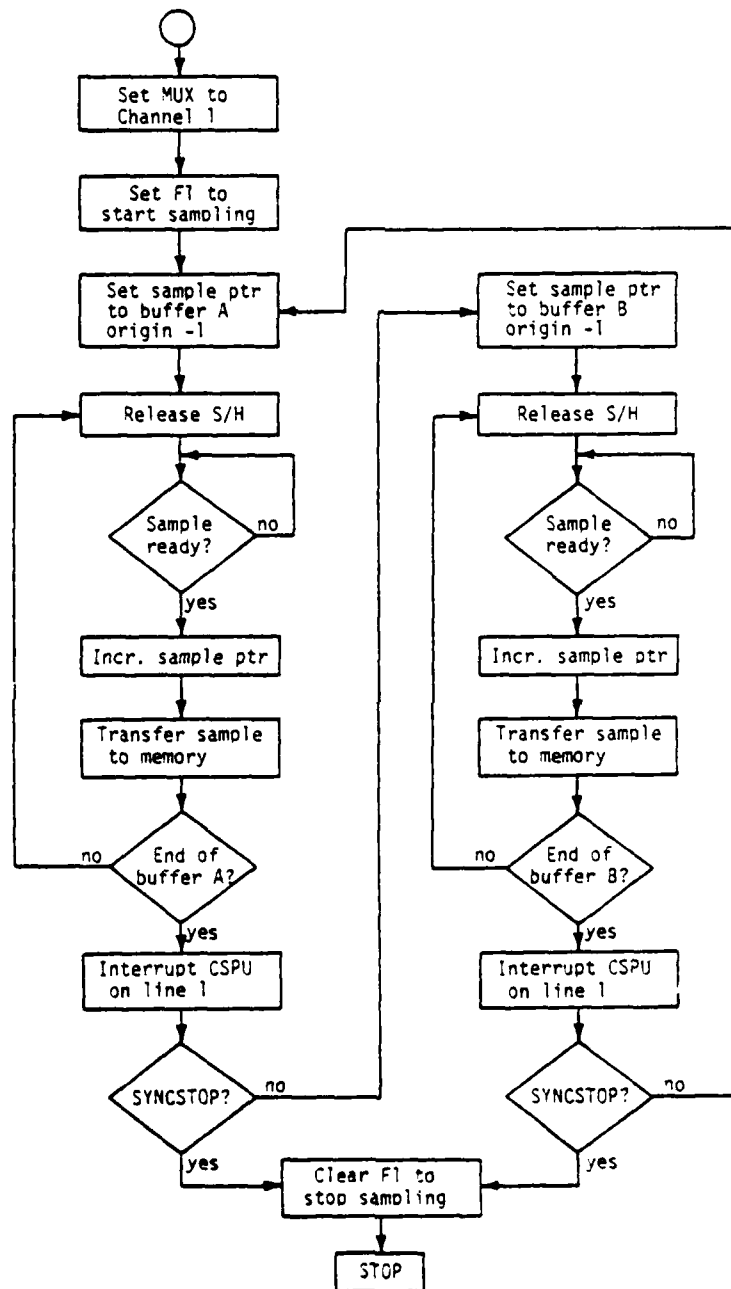


FIG. 5. ADPROG: ADAM SCROLL PROGRAM

2.2.1.2 A/D Interrupt Service (ADAMINT)

ADAMINT is activated by each ADAM line 1 interrupt, signifying the filling of another A/D input buffer. ADAMINT, like the other three speech coder input/output interrupt service routines, maintains a pair of integer scalar "pointer offsets" to keep track of its input and output buffer relationships. These pointer offsets, which take on the values -2 and 0, are used to reference small tables of address pointers. Thus, for example, ADPO ("A/D Pointer Offset") references ADBPTR, a table of pointers to the two A/D input buffers, and TSRPO ("TSOURCE Pointer Offset") references the tables TSRBPTR and TSRFPTR, which point to TSOURCE buffer-copying subroutines and buffer-status flags.

The operation of ADAMINT is shown in Fig. 6. When it receives a buffer-filled (line 1) interrupt, ADAMINT switches ADPO to the buffer just filled by the ADAM. Then ADAMINT checks the status of the next TSOURCE buffer to be filled; if it is available (empty) and the handset hookswitch is up (indicating that the handset is in service), ADAMINT copies the new A/D buffer to the current TSOURCE buffer. If the TSOURCE buffer is empty but the hookswitch is down, a preconstructed "tone" buffer is copied instead to the TSOURCE buffer. (This simulated input is intended to indicate to the person using the remote speech coder that this coder is in operation, but the handset is resting in the holder.) In either case, the TSOURCE buffer flag is set

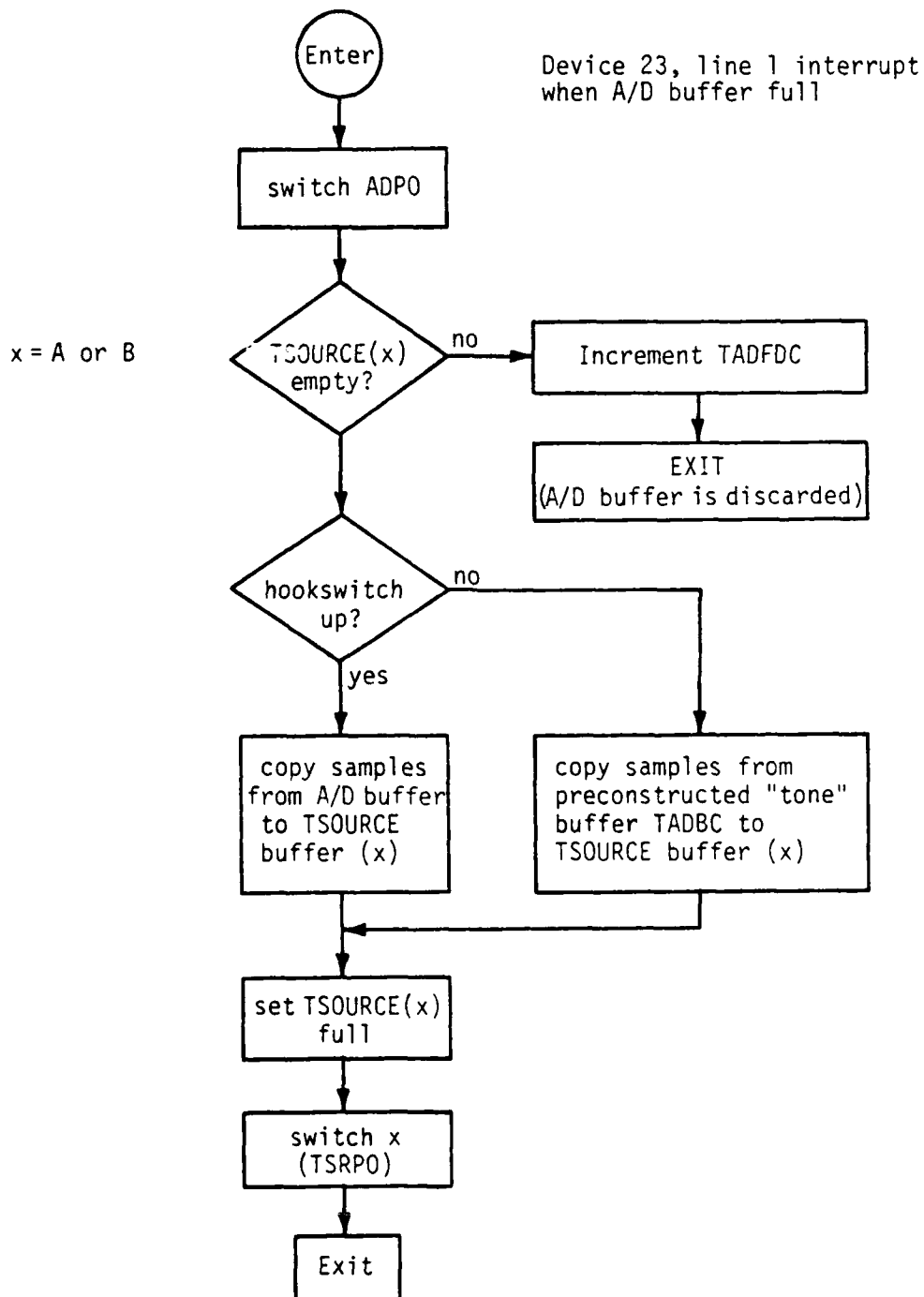


FIG. 6. A/D INTERRUPT SERVICE ROUTINE

to nonzero to indicate full, TSRPO is switched, and the routine exits. If, on the other hand, the TSOURCE flag shows that the current TSOURCE buffer is not empty and therefore unavailable, ADAMINT simply exits, effectively discarding the new A/D data. An "A/D Frame Discard Counter" (TADFDC) is incremented by one to keep track of the fact that an A/D buffer was discarded.

2.2.1.3 ANALYZE Module

The ANALYZE Module implements the APC analysis algorithm described in Section 2 of the "Specification of the Optimized 16 kb/s APC Algorithm" (henceforth known as the "Algorithm Specification") (Volume I, Appendix A of this report). As described in Section 2.1.3, two versions of the ANALYZE Module exist, one for each of the two sets of input/output buffers required for double buffering. Each version consists of a function list of MAP-300 function calls. The two versions (ANLZA and ANLZB) are identical in terms of the sequence of functions called and differ only in the parameters (buffers and/or scalars) passed to several of the functions.

This section describes, on a function by function basis, the ANLZA/ANLZB function lists, which appear in subroutine BBN16F of the speech coder software (Appendix D). The specification of alternative parameters for the two function lists is indicated in the descriptions below by a '/' (e.g., 'TSRA/B' indicates TSRA for the ANLZA function list and TSRB for the ANLZB function

list). Each function is either supplied by CSPI as part of the SNAP-II Release 3.5 software system or written by BBN. CSPI-supplied functions are described in detail in CSPI documents [2,3,4]. BBN-written functions are described in detail in Appendix A of this report. The buffers and scalars passed as parameters to the functions are defined in subroutine BBN16C of the speech coder software (Appendix D) and are described in detail in Appendices B and C.

2.2.1.3.1 Begin Function List

MPBFL(ANLZA/B)

This CSPI function specifies the start of the indicated function list.

2.2.1.3.2 Set G-flag

MPGSC(IG3,ISET)

This BBN function causes general-purpose flag G3 to be set, indicating the start of ANLZ processing. This use of flag G3 is intended for system debugging, timing, and internal measurement and is not required for proper operation of the speech coding software.

2.2.1.3.3 Monitor Peak Input Level

ADPEAK(TSRA/B)

This BBN function serves to maintain a running maximum (in integer scalar TADPK) of the digitized speech input samples, allowing the user to properly adjust the analog input signal level for maximum A/D range without

clipping. The maximum sample in the current input buffer is compared to the previous maximum, and TADPK is updated if necessary. TADPK is displayed (and reset to zero) during speech coder operation in response to a "T" command.

2.2.1.3.4 Transmitter Frame History Updates

THIST(TSP1,TSP0,TE1,TE11,TRH,TRH0)

This BBN function performs frame history updates on three buffers. The last-frame data from the end of each buffer is moved to the beginning of the buffer for use during current frame processing. (The buffers are referenced by TSP1, TE1, and TRH. The end portions of these buffers are referenced by TSP0, TE11, and TRH0.)

2.2.1.3.5 Preemphasis

DPRE(TSP0,TALPH,TSRA/B,TPHST,TONES)

This CSPI function (Discrete Weighted Pre-emphasis Filter) preemphasizes the input speech (TSRA/TSRB) (as indicated in Section 2.1 of the Algorithm Specification), with filter coefficient TALPH, filter history TPHST, unity weighting vector TONES, and output preemphasized speech TSP0.

2.2.1.3.6 Done with Input

MPWT(PCSR,AP)
MPIST(TSRFA/B,0)

These CSPI functions clear the appropriate buffer status flag (TSRFA or TSRFB) once the previous function has completed, indicating that the associated buffer can now be considered empty (i.e., ready to be filled with new input speech data).

2.2.1.3.7 Pitch Filter Analysis

The functions in this section perform pitch filter analysis on the preemphasized input speech, as indicated in Section 2.2 of the Algorithm Specification.

SSUM(TDCN,TSP,TFSZI)

This CSPI function sums the extended frame of preemphasized input speech samples (TSP), and multiplies the sum by the negative reciprocal of the extended frame size (TFSZI), producing the negative of the extended frame DC value (TDCN).

VMUL(TWSP,1,TSP,TDCN,THAM0,0)

This CSPI function removes the DC term (TDCN) from the input speech (TSP) and multiplies the result by a Hamming window (THAM0), producing windowed speech in TWSP.

FFTLR(TXSP,1,TWSPZ,TCOST,TXSP)

This BBN function is a slightly modified version of the CSPI function FFTNR (Real to complex FFT, not in place) (described in [4]). The modification involves a reformatting of the complex frequency domain output (TXSP) such that the last real output point appears at the end of the array, rather than in the first imaginary output location (which is known to be zero). The real input (TWSPZ) is a zero-padded copy of the windowed speech in TWSP. TCOST is a pregenerated table of cosine values.

VMSQ(TPSPR,TXSPR,TXSPI)

This CSPI function computes the power spectrum (TPSPR) of the windowed speech by computing the magnitude squared from the DFT real (TXSPR) and imaginary (TXSPI) parts.

VCLR(TPSPI)

This CSPI function clears the imaginary part of the complex buffer containing the real power spectrum. This clear operation must be performed every frame because the following inverse FFT function writes to TPSPI.

FFILR(TRP,1,TPSP,TCOST,TRP)

This BBN function is a slightly modified version of the CSPI function FFINR (Complex to real inverse FFT, not in place) (described in [4]). The modification involves reformatting of the complex frequency domain input (TPSP) corresponding to the modification described above for FFTLR. TPSP is the complex input consisting of real part TPSPR and imaginary part TPSPI. The real output TRP contains the autocorrelation coefficients of the windowed speech.

PITCH(TRPP,TMH,TRP,TIM,T14,TC2P)

This BBN function computes and codes the pitch lag (in terms of number of speech samples) from the autocorrelation coefficients (TRP). The pitch lag is output in TMH, and the coded pitch is output as an integer half-word in the left half of real scalar TIM. -TRP(pitch lag -1), -TRP(pitch lag), and -TRP(pitch lag +1) are output (for the three-tap pitch predictor coefficient computation) in TRPP. The single-tap pitch predictor coefficient is output in real scalar TC2P.

MEWL(TC,TINC,TRPP,TKTHR,TRP,TWORK)

This CSPI function (described in [4]) computes the three-tap pitch predictor coefficients (TC) from the pitch-centered autocorrelation coefficients (TRPP).

STAB(TCH,TC2P,TIC,TC)

This BBN function performs a stability check on the three-tap pitch predictor coefficients (TC) and uses the

single-tap predictor (with zeros for the first and third coefficients, TC2P as the second coefficient) if the three-tap coefficients are found to be unstable. The pitch predictor coefficients are then coded and quantized into buffers TIC and TCH, respectively.

2.2.1.3.8 Inverse Pitch Filter

INVPF (TE10, TMH, TSPl, TCH)

This BBN function inverse filters the preemphasized speech (TSPl) (as indicated in Section 2.3 of the Algorithm Specification), using quantized pitch predictor coefficients (TCH) and pitch lag (TMH) and produces a first residual signal (TE10).

2.2.1.3.9 Spectral Filter Analysis

The functions in this section perform spectral filter analysis on the first residual signal, as indicated in Section 2.4 of the Algorithm Specification.

VMUL (TWE1, 1, TE10, 0, THAM1, 0)

This CSPI function multiplies the first residual signal (TE10) by a Hamming window (THAM1), producing a windowed first residual (TWE1).

DCORZ (TRS, 0, TWE1, TWE1)

This CSPI function (described in [4]) computes seven autocorrelation coefficients (TRS) from the windowed first residual (TWE1).

HFC (TRSP, TRS, TLMU)

This BBN function modifies the autocorrelation coefficients (TRS), producing high-frequency-corrected autocorrelation coefficients (TRSP).

MWLQ6 (TKE,TKTHR,TKH,TRSP,TIK)

This BBN function is a modified version of the CSPI function MWLD. It uses the Levinson recursion to compute the spectral reflection coefficients from the corrected autocorrelation coefficients (TRSP), and then codes and quantizes them into buffers TIK and TKH.

2.2.1.3.10 Inverse Spectral Filter

The functions in this section perform inverse filtering of the first residual signal, as indicated in Sections 2.4.5 and 2.5 of the Algorithm Specification.

VKTOA (TAH,TKH)

This BBN function converts reflection coefficients (TKH) to spectral predictor coefficients (TAH).

DCORZ (TE2,0,TE1,TAHR)

This CSPI function (described in [4]) inverse filters the first residual (TE1) by performing a correlation operation with the pitch predictor coefficients in reverse order (TAHR) (hence a convolution), producing a second residual signal (TE2).

2.2.1.3.11 Error Protection on Previous Frame Parameter Data

PROPAR (B/A)

This BBN function performs error protection on previous frame parameter data (in buffer TSNKB/TSNKA), depositing error protected and bitstreamed transmission data in buffer TBTA/TBTA. This function executes in the CSPU while the previous function (DCORZ) is executing in the AP. (See Section 2.2.1.3.18 for a more detailed discussion of this module.)

2.2.1.3.12 Noise Shaping Filter Calculation

VMUL(TANS,1,TAH,0,TFAC,0)

This CSPI function computes noise shaping filter coefficients (TANS) as indicated in Section 2.4.6 of the Algorithm Specification.

2.2.1.3.13 Gain Factor Calculation

GAIN(TGFAC,TIG,TIDG,T72R,TE2,T3R)

This BBN function computes and quantizes energy and segment delta-gains as indicated in Section 2.6 of the Algorithm Specification. TIG is the coded energy of the second residual (TE2). TIDG contains the three coded segment delta-gains, and TGFAC contains the three residual quantizer segment scale factors.

The quantization table for energy differs from that in the Algorithm Specification by a scale factor of $2.511 \times (10^{-7})$ (-66 dB), which accounts for the difference in input speech signal magnitude between the simulation (+1024) and the real-time implementation (+0.5). (The ADAM and AOM have a range of ± 1 , but that amplitude is rare.) The energy and delta-gain decoding tables differ from their quantization tables by a square root. Quantization is performed on the computed energies, but the APC and synthesis operations use gain (or square root of energy) factors. The gain decoding table also differs from that in the Algorithm Specification by -66 dB (scale factor of $5.01 \times (10^{-4})$).

2.2.1.3.14 APC Residual Calculation

APC(TIUA/B,TMH,TSP0,TFILT,TGFAC,TRH,TVH,TQ1)

This BBN function computes and codes the APC residual as indicated in Section 2.7 of the Algorithm Specification. The input preemphasized speech is in TSP0. The spectral, pitch, and noise shaping filter coefficients are contiguous in buffer TFILT. TMH

contains the pitch lag. TRH, TVH, and TQ1 retain frame-to-frame memories of the filters. The coded APC residual is stored in buffer TIUA/TIUB (equivalent to the beginning of TSNKA/TSNKB).

2.2.1.3.15 Error Protection on Previous Frame Residual Data

PRORES(B/A)
MPIST(TBTFB/A,1)

These functions (PRORES is a BBN function; MPIST is a CSPI function) perform bitstreaming on the previous frame's residual data (in buffer TSNKB/TSNKA), depositing bitstreamed transmission data in buffer TBTB/TBTA. The appropriate buffer status flag (TBTFB or TBTFA) is set, indicating that the associated buffer is now full (i.e., ready to be transmitted). These functions execute in the CSPI while the previous function (APC) is executing in the AP. (See Section 2.2.1.3.18 for a more detailed discussion of this module.)

2.2.1.3.16 Transmitter Data Collection

GTHR(TSNKA/B,TIG,TIDG,TIM,TIC,TIK)

This BBN function collects all coded parameter transmission data into a single buffer (TSNKA/TSNKB). (The coded residual data has been placed directly into TSNKA/TSNKB by the APC function).

2.2.1.3.17 End of Function List

MPEFL(ANLZA/B)

This CSPI function specifies the end of the indicated function list.

2.2.1.3.18 Error Protection and Bitstreaming

The PROPAR and PRORES modules take as input a TSINK buffer

(either TSNKA or TSNKB) containing quantized and coded analysis parameters and residual samples and produce as output a TBITS buffer (again, TBTA or TBTB according to the TSINK buffer designation) in which:

1. certain high-order data bits have been grouped together and protected with (7,4) Hamming codewords;
2. the data to be transmitted has been "bitstreamed", one bit per half-word, in the form used by the TMODEM scroll program;
3. histogram information of the coded analysis parameters has been recorded.

These operations are performed in the CSPU, and therefore they can be executed concurrently with an AP operation. These operations are implemented as two separate CSPU modules so that each module's execution time is (mostly) hidden by that of an AP module. The PROPAR module performs the error-protection and bitstreaming of the analysis parameters (gain, delta-gains, pitch, pitch filter coefficients, and reflection coefficients), and it also records the histogram information for those parameters (except pitch). PRORES bitstreams the residual values only.

The format of the TSINK buffer is shown in Table 1, along with the number of bits per parameter and the number of high-order bits protected by Hamming codes. (C1-C3 denote the three pitch filter coefficients, and K1-K6 denote the six reflection coefficients.) A total of 44 high-order bits of the transmitted parameters are protected using 11 Hamming (7,4) codewords.

<u>Word</u>	<u>Parameter</u>	<u>No. of bits</u>	<u>Bits protected</u>
0-215	Residual(216 samples)	2	0
216	Gain	6	6
217-219	Delta-gain (3)	2	2
220	Pitch	7	7
221	C1	3	2
222	C2	4	3
223	C3	3	2
224	K1	6	5
225	K2	5	4
226	K3	4	3
227-229	K4,K5,K6	4	2

TABLE 1. TSINK BUFFER FORMAT

The TBITS buffer contains one data bit in the rightmost bit of each 16-bit half-word. The format of the TBITS buffer (also the frame of data transmitted) is shown in Table 2. (Bits of coded parameters are numbered starting with bit 0 on the right. B5432 thus denotes a 4-bit field of bits 5 through 2.)

A histogram-gathering function was included in the PROPAR module for gathering statistics on the effectiveness of the quantization tables for the analysis parameters and for verifying correct transmitter operation. A PDP-11 program (HIST16) was implemented (not delivered, but shown in Appendix D) to read the histogram buffers defined on Bus 1 and list them in a text file.

2.2.1.4 TMODEM Interrupt Service (TMODEMINT)

The TMODEM Interrupt Service routine is shown in Fig. 7. When activated by a Line 1 interrupt from the modem IOS-2 scroll, this routine updates the TMODEM pointer offset to point to the

<u>Bit No.</u>	<u>Function</u>
0	Sync bit
1-2	1st residual sample (LSB first)
.	.
.	.
431-432	216th residual sample
433-439	B5432 gain (Hamming (7,4) codeword)
440-446	B10 gain, B10 delta-gain-1 (7,4)
447-453	B10 delta-gain-2, B10 delta-gain-3 (7,4)
454-460	B6543 pitch (7,4)
461	B0 C1
462-468	B2 C1, B210 pitch (7,4)
469	B0 C2
470-476	B321 C2, B1 C1 (7,4)
477	B0 C3
478	B0 K1
479-485	B54 K1, B21 C3 (7,4)
486	B0 K2
487-493	B4321 K2 (7,4)
494	B0 K3
495-501	B3 K3, B321 K1 (7,4)
502-508	B32 K4, B21 K3 (7,4)
509-510	B10 K4
511-512	B10 K5
513-514	B10 K6
515-521	B32 K6, B32 K5 (7,4)

TABLE 2. TBITS BUFFER FORMAT

current (just emptied) TMODEM buffer, then checks the current TBITS buffer flag to see if there is new bitstream data ready to be transmitted. If the flag is set to full, the data is copied from the TBITS buffer to the current TMODEM buffer, the TBITS flag is set to empty, and the TBITS pointer offset is updated to point to the other TBITS buffer/flag. If, on the other hand, the current TBITS buffer flag indicates not-full, bitstream data corresponding to a frame of silence is copied (from buffer TBTC) to the current TMODEM buffer, the "fake frame" is counted by

incrementing TMFFC, and the routine exits without having changed the TBITS pointer offset.

In either case, only 521 words (bits) of data are copied to the TMODEM buffer. The data bit of the first word of each TMODEM buffer is not copied into, as it is the synchronization bit: a 0 in TMODEMA and a 1 in TMODEMB. Since data is transmitted alternately from TMODEMA and TMODEMB, the data frames at the receiver have the sync bit alternating between 0 and 1 in successive frames.

2.2.1.5 TMODEM and RMODEM I/O Scroll Program (RTPROG)

The digital data input and output is performed by an IOS-2SM Scroll processor that has been augmented by a modem interface. This interface also contains the dual-rate clock for timing both the modem data and the speech sample input and output.

The TMODEM and RMODEM Scroll program (RTPROG) is shown in Fig. 8. Although the TMODEM output and RMODEM input processes are logically distinct, they are performed by this single program. After setting the rates of the modem-data and speech-sample clocks and initializing buffer switches and address pointers, the program enters a basic loop that checks the two peripheral flags for data ready from the modem receiver interface (P1 set) or for the modem transmitter interface ready to accept data (P2 set).

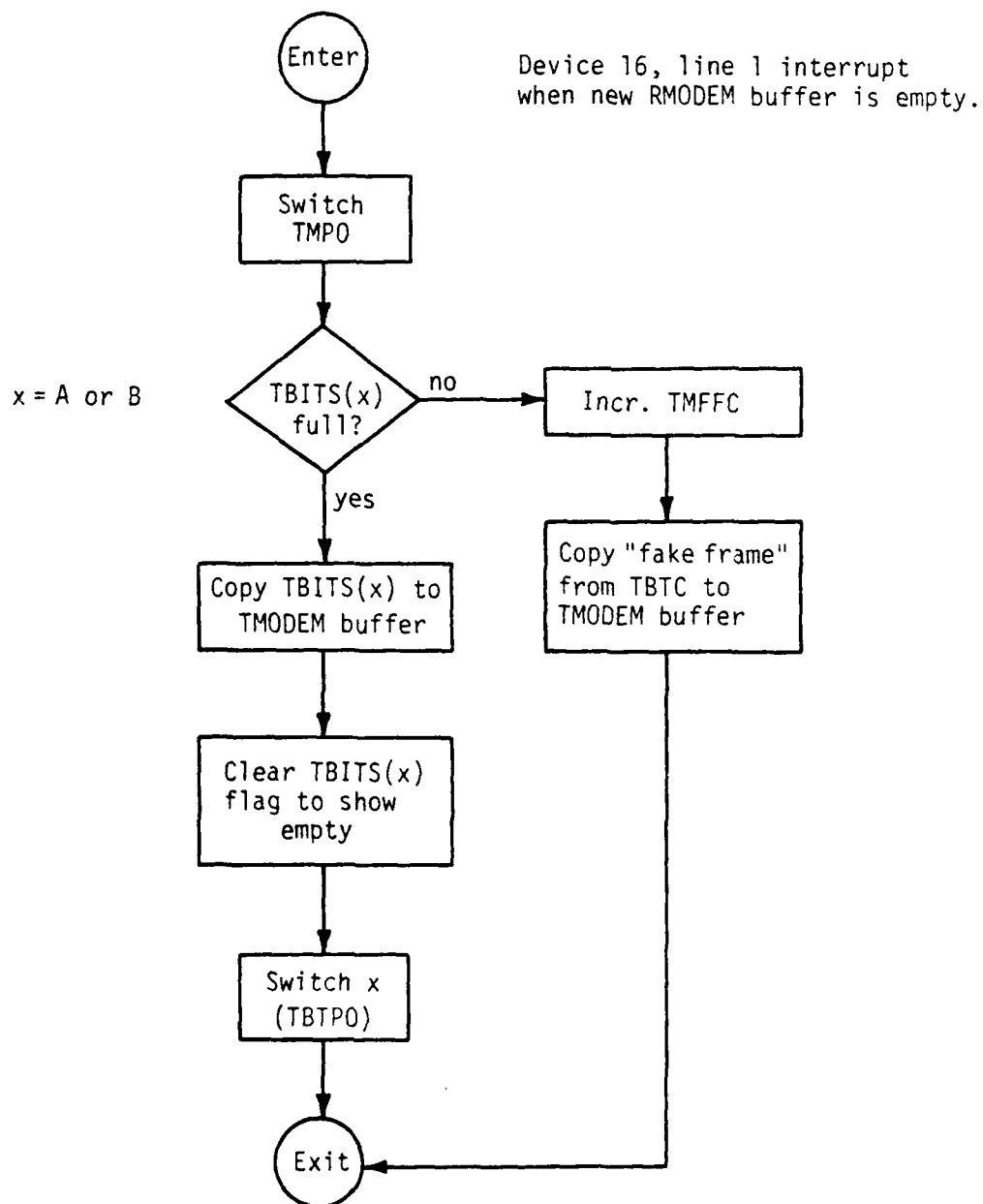


FIG. 7. TMODEM INTERRUPT SERVICE ROUTINE (TMODEMINT)

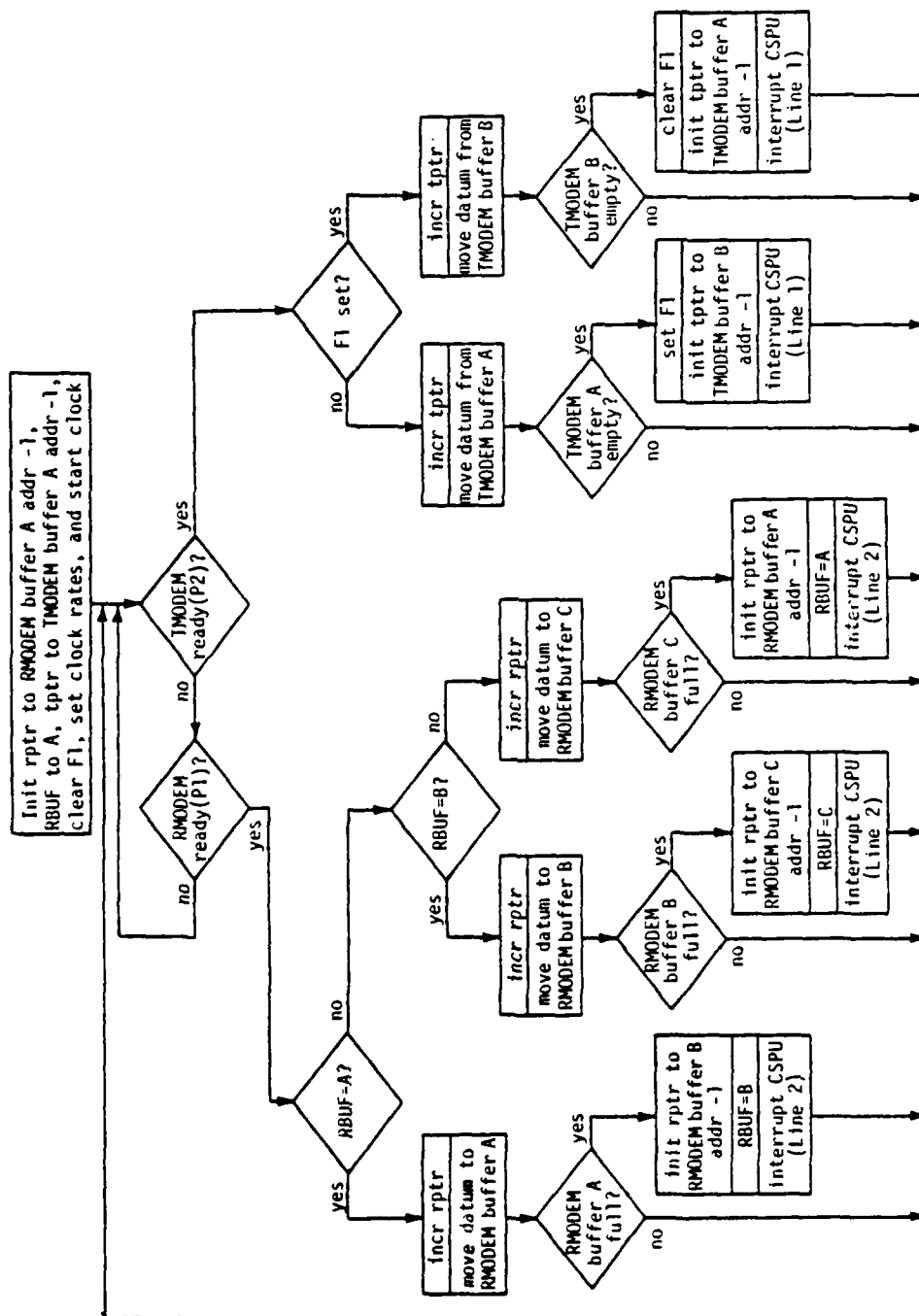


FIG. 8. TMODEM AND RMODEM SCROLL PROGRAM FLOW CHART

When a new datum is available from the interface, it is transferred to RMODEM buffer A, B, or C, depending on the value of a buffer switch in register R0 and an address pointer in register R1. The transfer clears flag P1. If the input buffer is now full, the buffer switch and address pointer are changed to point to the next buffer and the CSPU is interrupted on line 2.

When the interface becomes ready to accept a datum (to be transmitted), a word is transferred from TMODEM buffer A or B, depending on the value of flag F1, used as a buffer switch, and an address pointer in register R2. The transfer clears flag P2. If the output buffer is now empty, the buffer switch and address pointer are changed to indicate the other TMODEM buffer and the CSPU is interrupted on line 1.

2.2.1.6 Transmitter Input/Output Buffer Initial Sequence

The several layers of shared input/output buffers in the speech coder transmitter and receiver require that the modules that reference them be initialized to do so in the proper order. The start-up sequence for the transmitter is described below.

The ADAM scroll program starts by writing speech samples into buffer TADBA (and then proceeds to write into TADBB, etc.). At the first ADAM interrupt, ADAMINT copies from TADBA to TSRA (TSOURCE A buffer) and sets the flag TSRFA. The background process executes ANLZA first, since TSRFA=1 and TBTFB is initially 0. The execution of ANLZA includes PROPAR(B) and

PRORES(B), which write the first bitstream data into buffer TBTB; then the flag TBTFB is set to 1. The TMODEMINT interrupt routine is initialized to expect its first bitstream data in buffer TBTB, so that is the first data copied into a TMODEM buffer for output by the TMODEM scroll program.

Meanwhile, the TMODEM scroll program has started by transmitting (initialized) data from buffer TMDMA. It then sends an interrupt and proceeds to read from TMDMB, and so forth. The initial TMODEM interrupt activates TMODEMINT, whose first task is to provide new data for the just-emptied TMDMA buffer. TMODEMINT is initialized to check the flag TBTFB, as described above. At the end of the first frame, however, ANLZA has not yet executed and there is as yet no data in TBTB, so TMODEMINT copies "dummy" bitstream data instead from TBTC into TMDMA and exits. The next time TMODEMINT is activated, it again checks TBTFB and (since ANLZA has run by now) finds it set, so it copies the new bitstream data from TBTB to the (just-emptied) TMDMB buffer. Thereafter all data buffers are alternately filled and emptied without conflict.

2.2.2 Receiver

The Receiver is similar in structure to the Transmitter. The Receiver is shown in Fig. 9.

Data is accepted from the modem and put into one of three

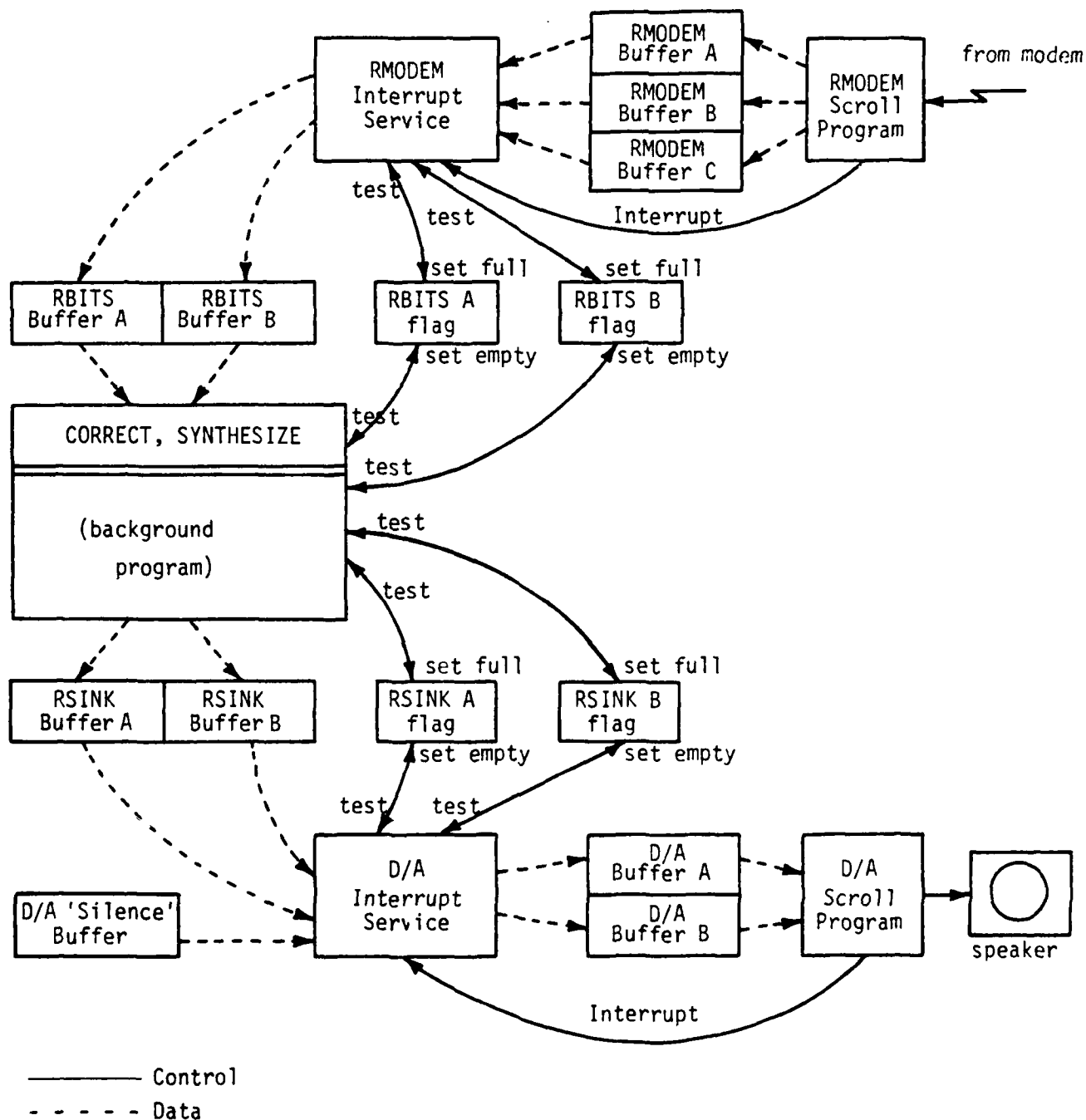


FIG. 9. RECEIVER PROCESS

RMODEM buffers by the RMODEM program running in an I/O scroll. When this program fills a buffer, it generates an interrupt to the CSPU and begins filling the next buffer (A,B,C,... in rotation).

This interrupt activates the RMODEM Interrupt Service routine, which checks frame synchronization and, if correct, transfers the new data to an empty RBITS buffer and sets the corresponding RBITS flag to indicate full. The data that is actually transferred does not correspond exactly to the just-filled RMODEM buffer. Rather, it is a "frame" of data completely contained in the new RMODEM buffer and the RMODEM buffer previously filled. A frame of data begins with a sync bit. Since this bit may occur anywhere within a single RMODEM buffer, two full buffers (and therefore three buffers altogether) are required to guarantee that a complete frame is available.

The CORPAR and DECODA/B modules, running in the CSPU at background level, empty the full RBITS buffer, perform error correction and decoding of the incoming bitstream, copy these decoded parameters into an (empty) RSOURCE buffer, and clear the RBITS flag to signify empty.

The SYNTHESIZE module, running in the CSPU at background level, empties the (full) RSOURCE buffer, performs the processing necessary to synthesize speech, and puts the output speech samples into an (empty) RSINK buffer, setting the corresponding

RSINK flag to indicate full. The SYNTHESIZE module, although logically following the CORPAR and DECODA/B modules, in fact contains them and is executed concurrently with them. That is, SYNTHESIZE operates on the data supplied by the previous execution of CORPAR and DECODA/B. The first time that SYNTHESIZE is executed, it operates on a RSOURCE buffer that has not been filled by CORPAR and DECODA/B; therefore the RSOURCE buffers are initialized to contain data that will synthesize silence.

The data from the RSINK buffer is transferred to an empty D/A buffer by the D/A Interrupt Service routine. This routine also sets the corresponding RSINK flag to indicate empty.

This Interrupt Service routine is activated by an interrupt from the D/A Scroll program, running in another I/O scroll (the AOM, or Analog Output Module). This scroll program transfers data from a D/A buffer to the D/A converter, interrupts the CSPU when the buffer is empty, and begins transferring data from the other D/A buffer.

2.2.2.1 RMODEM Scroll Program

The RMODEM scroll program is described in Section 2.2.1.5.

2.2.2.2 RMODEM Interrupt Service (RMODEMINT)

The RMODEM Interrupt Service routine is shown in Fig. 10. It is activated by a line 2 interrupt from the modem IOS-2 scroll, which indicates that the next RMODEM buffer is full. The

routine updates the RMODEM pointer offset to point to the just-filled buffer. Then it copies the on-hook bit from the first word in the current RMODEM buffer to the integer scalar RONHK, where it is saved for use by the ADAM interrupt service routine (Section 2.2.1.2).

Then the integer scalar RERRS is tested. This is normally zero, but if it is set to nonzero (by a command from the host program), a channel error simulation routine (RMI\$SIM) is called. RMI\$SIM uses the value in RERRS as the number of errors to be introduced (quasi-randomly) per frame. Since there are 522 bits per frame, each error contributes about 0.2% to the error rate. This error simulation is intended only for demonstration; it is not an accurate simulation of an errorful channel.

The next set of operations deals with frame synchronization. Since the receiver has no prior knowledge about the position of the frame boundaries in the input data stream, it must infer the boundary position from the received data pattern. The first bit of each transmitted frame is the sync bit, which alternates between 0 and 1 in successive frames; this pattern allows the receiver to detect and maintain frame synchronization. The synchronization routines are described in Section 2.2.2.2.1 below.

RMODEMINT tests the value of RSYNC, an integer scalar that maintains the receiver's sync-state. If RSYNC=0 (as it is when

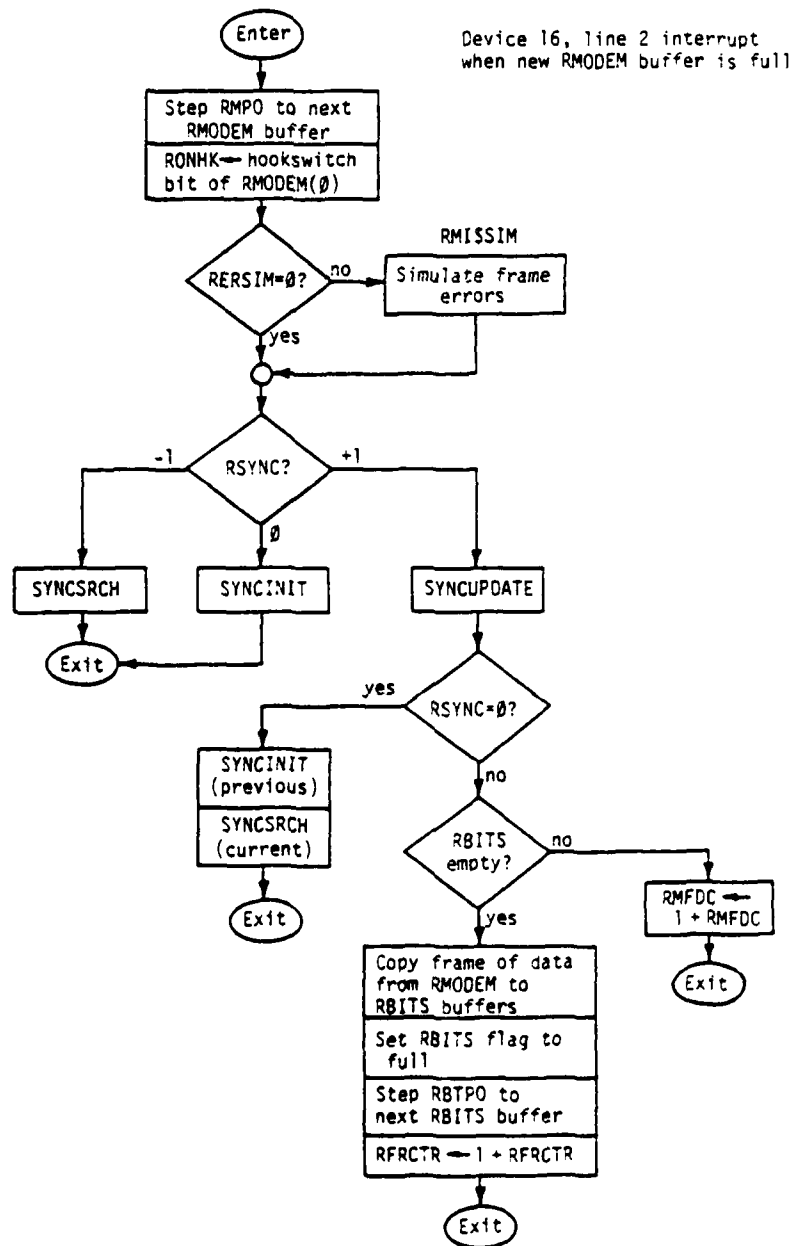


FIG. 10. RMODEM INTERRUPT SERVICE MODULE (RMODEMINT)

the speech coder is initialized), nothing is known about sync, so the SYNCINIT routine is called to initialize the sync-search process with the current RMODEM buffer. If $RSYNC < 0$, then the receiver is still searching for sync, so the SYNCSEARCH routine is called to continue the search using the new data in the current RMODEM buffer. In both of these cases, RMODEMINT exits immediately. If $RSYNC > 0$, then the receiver has gained frame-sync, so the SYNCUPDATE routine is called to check that the current RMODEM buffer continues to show the expected sync-bit value. Upon the return from SYNCUPDATE, RSYNC is checked again; if it has been set to zero, then the receiver has lost sync and must again search the incoming bitstream for the sync-bit pattern; therefore SYNCINIT and SYNCSEARCH are called to restart sync-searching with the previous and current RMODEM buffers.

If RSYNC is still greater than zero after SYNCUPDATE, then synchronization is confirmed, and the receiver makes use of the data for speech output. RMODEMINT checks the current RBITS buffer flag; if empty, the most recent frame of data is copied from the RMODEM buffers to the current RBITS flag, the RBITS flag is set to full, and the RBITS pointer offset (RBTP0) is switched to the next RBITS buffer. Note that in general, the current frame will straddle the previous and current RMODEM buffers. It is for this reason that there are three RMODEM buffers, two to hold the current frame while the third is being filled by the RMODEM scroll program.

If, for some reason, the RBITS buffer is not empty (ready to accept new data), the new data frame is effectively discarded. A frame discard counter (RMFDC) is incremented to take note of this, and the routine exits without updating RBTPO.

2.2.2.2.1 Synchronization Routines

The SYNCINIT subroutine (Fig. 11) initializes the state of the frame synchronization section of the speech coder.

Its primary function is to copy the data bit of each word of the current RMODEM buffer into RSSPF, the "previous frame" memory used in SYNCsrch, and to clear the counts in the RSSSS buffer. SYNCINIT also zeros the gain word of both RSOURCE buffers and the entire previous RBITS buffer, so that when the speech coder receiver resumes operation after regaining sync, the incorrect information in these buffers will not produce unwanted transients in the speech output.

The SYNCsrch subroutine (Fig. 12) scans the incoming RMODEM buffers (one per call), building up statistics on the frame-to-frame data patterns in each bit position of the frame until it detects a large enough number (ACQTHR) of consecutive bit alternations in a single bit position to allow it to identify that position as carrying the sync bit.

When that occurs, it sets the value of RBOFO ("beginning-of-frame-offset") to the distance from the start of the buffer to

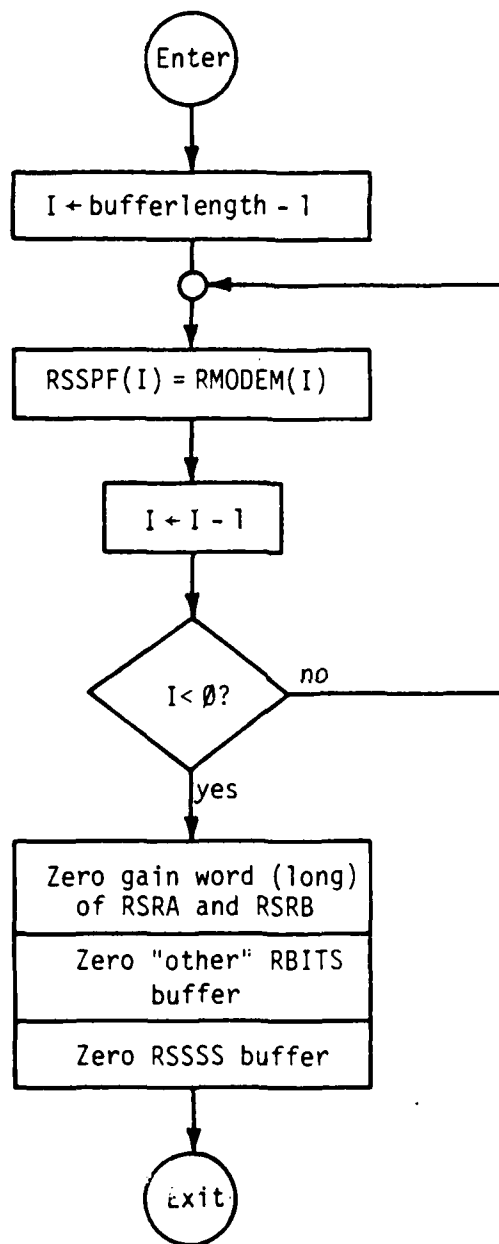


FIG. 11. FRAME SYNCHRONIZATION INITIALIZATION MODULE (SYNCINIT)

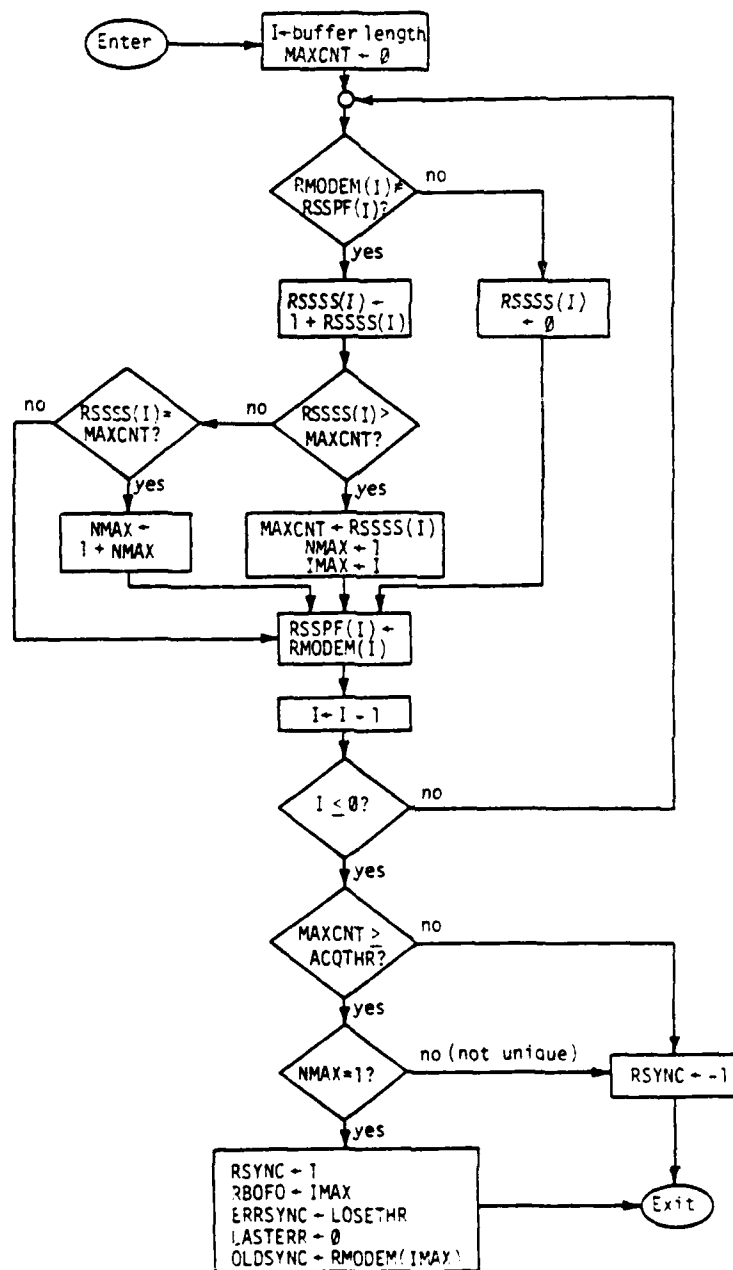


FIG. 12. FRAME SYNCHRONIZATION SEARCH MODULE (SYNCSRCH)

the sync-bit position and RSYNC to +1 to cause RMODEMINT to start passing input frames to the speech coder synthesizer on the next RMODEM interrupt. SYNCSEARCH also initializes three variables that are used in subsequent calls to SYNCUPDATE.

The SYNCUPDATE subroutine (Fig. 13) is used once sync has been acquired, to check the sync-bit position of each incoming RMODEM buffer to see if it has the expected value.

It is the function of SYNCUPDATE to declare sync lost if the declared sync bit does not show the expected frame-to-frame alternation. Since channel errors may cause the sync bit to be incorrect, the loss-of-sync criterion must be more stringent than a single incorrect sync bit. The criterion is a sufficient number (LOSETHR) of incorrect sync bits without two consecutive correct sync bits. A simpler criterion, such as LOSETHR consecutive sync bit errors, would be unsatisfactory, since the high-order bit of many speech coder parameters (e.g., energy) may have the same value for many consecutive frames, and a constant bit would compare "correctly" with an alternating bit sequence every other frame.

The performance of the synchronization routines is controlled by two thresholds, ACQTHR and LOSETHR. It is necessary to set ACQTHR high enough so that it is very unlikely that SYNCSEARCH will detect sync on the wrong bit position. The value of LOSETHR must be high enough that SYNCUPDATE is unlikely

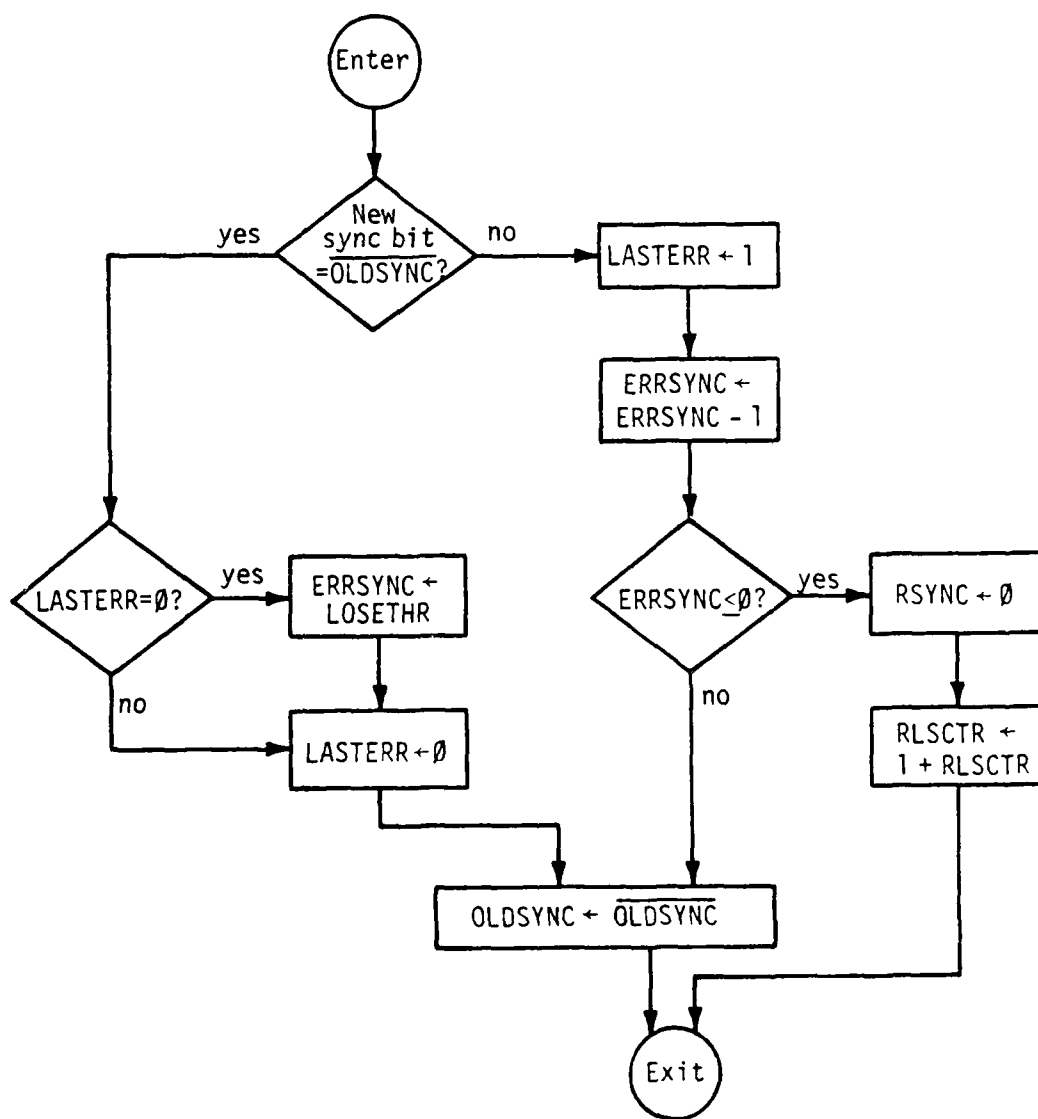


FIG. 13. FRAME SYNCHRONIZATION UPDATE MODULE (SYNCUPDATE)

to declare loss-of-sync due to channel errors falling on the sync bit, but small enough that if sync should be lost for some reason, the receiver does not synthesize too much "garbage" before detecting the loss-of-sync. Of course, while the receiver is searching for sync, RMODEMINT does not pass data to the synthesizer, so silence is output. For a 1% channel bit error rate, the values of ACQTHR=10 and LOSETHR=4 give a probability of acquiring false sync of about 0.0002 and an expected time to spontaneous loss-of-sync of about 10 hours, respectively.

2.2.2.3 SYNTHESIZE Module

The SYNTHESIZE Module implements the APC synthesis algorithm described in Section 3 of the Algorithm Specification. As described in Section 2.1.3, two versions of the SYNTHESIZE Module exist, one for each of the two sets of input/output buffers required for double buffering. Each version consists of a function list of MAP-300 function calls. The two versions (SYNZA and SYNZB) are identical in terms of the sequence of functions called and differ only in the parameters (buffers and/or scalars) passed to several of the functions.

As in the case of Section 2.2.1.3, this section describes, on a function by function basis, the SYNZA/SYNZB function lists, which also appear in subroutine BBN16F.

2.2.2.3.1 Unbitstreaming and Error Correction

The CORPAR and DECODA/B modules take as input an RBITS

buffer containing a frame of bitstream data input from the modem and produce as output an RSOURCE buffer containing error-corrected and decoded floating point values of analysis/synthesis parameters and residual samples, ready for immediate use by the speech coder synthesizer.

The format of the RBITS buffer is the same as the TBITS buffer, shown in Table 2 above. The data bits are regrouped, and the 7-bit codewords are error-corrected by lookup in an inverse Hamming code table. The bits are then grouped into coded parameter values and decoded by lookup in parameter-specific decoding tables of floating-point values. (As noted in Section 2.2.1.3.13, the gain and delta-gain decoding tables differ from the ones in the Algorithm Specification by a scale factor.) The format of the RSOURCE buffer is similar to that of the TSINK buffer (Table 1), except that the half-word coded TSINK values are replaced by full-word (32-bit) floating point values.

As in the case of the error-protection and bitstreaming operation, the unbitstreaming, error-correction, and decoding operation is performed by two separate modules, which are part of the SYNTHESIZE function list. CORPAR does the operations on the analysis parameters, while DECODA/B performs unbitstreaming and decoding on the residual samples. Unlike the other "bit-pushing" modules, DECODA/B is an AP module. (There are two distinct DECOD modules. DECODA decodes from buffer RBTA to buffer RSRA; DECODB

decodes using the other pair of buffers. The SNAP-II Executive permits the definition of only 63 distinct data buffers; this proved to be too few to allow the definition (to SNAP-II) of the RBITS buffers. Therefore, two DECOD modules for the APS were created, each with the parameters of its data buffers "hard bound" in the source code.)

2.2.2.3.2 Begin Function List

MPBFL(SYNZA/B)

This CSPI function specifies the start of the indicated function list.

2.2.2.3.3 Clear G-flag

MPGSC(IG3,ICLR)

This BBN function causes general purpose flag G3 to be cleared, indicating the start of SYNZ processing. This use of flag G3 is intended for system debugging, timing, and internal measurement and is not required for proper operation of the speech coding software.

2.2.2.3.4 Receiver Data Distribution

DEAL(RSRA/B, RGH, RDGH, RMH, RCH, RKH)

This BBN function distributes all decoded parameter data from a single buffer (RSRA/RSRB) to separate buffers and scalars. (The decoded residual data will be read directly from RSRA/RSRB by the SCLRES function.)

2.2.2.3.5 Receiver Frame History Buffer Update

VMOV(RRH,RRH0)

This CSPI function performs a frame history update on a single buffer. The last-frame data from the end of the buffer (RRH0) is moved to the beginning of the buffer (RRH) for use during current frame processing.

2.2.2.3.6 Residual Scaling

SCLRES(RWH,RGH,RDGH,RUHA/B)

This BBN function computes quantizer scale factors and scales the received APC residual (RUHA/RUHB are equivalent to the beginning of the RSRA/RSRB) into buffer RWH, as indicated in Sections 3.2 and 3.3 of the Algorithm Specification.

2.2.2.3.7 Spectral Synthesis

VLTSY(RVH,RVHM,RKH,RWH)

This BBN function performs a lattice-form spectral synthesis filtering operation on the scaled APC residual (RWH) as indicated in Section 3.5 of the Algorithm Specification, using reflection coefficients (RKH) directly. The output is in RVH. (This function implements an 8-pole filter; we are using it with the last two coefficients zeroed to implement a 6-pole filter.)

2.2.2.3.8 Decoding of Previous Frame Receiver Residual Data and Error Correction on Previous Frame Receiver Parameter Data

DECODEB/A
CORPAR(B/A)
MPWT(PRCR,AP)
MPIST(RBTFB/A,0)

These functions (DECOB/DECODA and CORPAR are BBN-written; MPWT and MPIST are CSPI-supplied) decode the previous frame received residual samples and error-correct the previous frame received parameter data (from RBTB/RBTA), depositing decoded data in buffer RSRB/RSRA. The appropriate buffer status flag (RBTFB or RBTFA) is cleared, indicating that the associated buffer can now be considered empty (i.e., ready to be filled with new received coded data). The last three function execute in the CSPU while the first executes in the AP. (See Section 2.2.2.3.1 for a more detailed discussion of DECODA/DECOB and CORPAR.)

2.2.2.3.9 Pitch Synthesis

PITSYN(RRH,RMH,RVH,RCH)

This BBN function performs a pitch synthesis filtering operation as indicated in Section 3.6 of the Algorithm Specification. The input is in RVH, and the output is in RRH.

2.2.2.3.10 Deemphasis

DFL22(RRHDA/B,RDCF0,RRH0,RDMY0)

This CSPI function does signal deemphasis as indicated in Section 3.7 of the Algorithm Specification. The filter coefficients are in four consecutive scalars starting with RDCF0. The filter memory is in four consecutive scalars starting with RDMY0. The synthesized speech is stored in RRHDA/RRHDB (equivalent to RSNKA/RSNKB).

2.2.2.3.11 Output Ready

MPWT(PCRSR,AP)
MPIST(RSNFA/B,1)

These CSPI functions set the appropriate buffer status flag (RSNFA or RSNFB) once the previous function has

completed, indicating that the associated buffer is full (i.e., ready to be emptied).

2.2.2.3.12 End of Function List

MPEFL(SYNZA/B)

This CSPI function specifies the end of the indicated function list.

2.2.2.4 D/A Interrupt Service Routine (AOMINT)

AOMINT is activated by each AOM line 1 interrupt, signifying the emptying of a D/A output buffer by the AOM scroll program. Its operation, as illustrated by Fig. 14, is analogous to that of the TMODEMINT routine, the only difference being the copying of synthesis output data from RSINK buffers to D/A buffers for output to the AOM. If an RSINK buffer is not available (full), a buffer of silence is output in its place. This happens whenever the speech coder is not receiving data from the remote speech coder and during frame synchronization or resynchronization.

2.2.2.5 D/A Scroll Program (DAPROG)

Speech output is performed by the AOM, an IOS-2 scroll processor that contains two D/A converters. The AOM gets its sample output clock from the SPI, and it sends the D/A output signal to the SPI for subsequent low-pass filtering and output. The signal is in the range -5 to +5 volts, and the sample rate is 6.621 kHz, the same as for the A/D input.

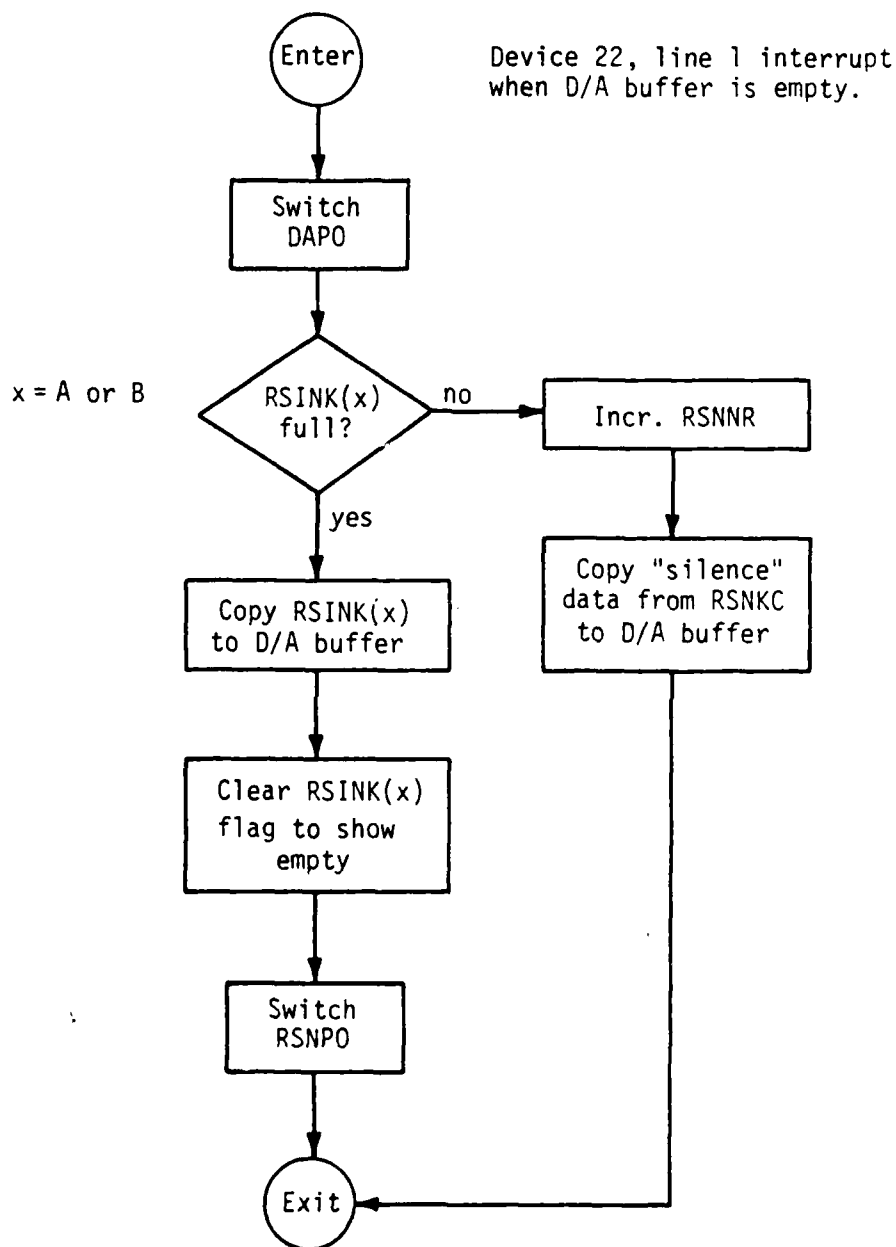


FIG. 14. D/A INTERRUPT SERVICE ROUTINE (AOMINT)

The AOM program (DAPROG) is illustrated in Fig. 15. In "single-channel" mode, the AOM outputs an internally-generated value on D/A Channel 0 while it converts speech samples from MAP memory on D/A Channel 1. The Channel 0 signal is a ramp that is reset each frame time; it is irrelevant to speech coder operation, but it may be used for synchronization or horizontal sweep on an external oscilloscope.

The operation of DAPROG is analogous to that of ADPROG. Output begins from buffer RDABA, and when the end of that buffer is reached, a line 1 interrupt is sent to the CSPU, and buffer RDABB is selected. Reaching the end of RDABB produces another line 1 interrupt and a switch to RDABA, and so forth.

2.2.2.6 Receiver Input/Output Buffer Initial Sequence

The buffer start-up sequence for the speech coder receiver is described below.

The RMODEM scroll program starts by writing received data words into buffer RMDMA, sending an interrupt when it is filled. Upon the initial interrupt, RMODEMINT begins by using RMDMA for frame-sync initialization. Since frame-sync has not yet been acquired, RMODEMINT does not output any data. Successive RMODEM interrupts cause RMODEMINT to use buffers RMDMB, RMDMC, RMDMA, etc. for further frame-sync searching until, after at least 12 input frames, sync has been found. Then RMODEMINT will copy a frame of bitstream data from the RMODEM buffers to buffer RBTA

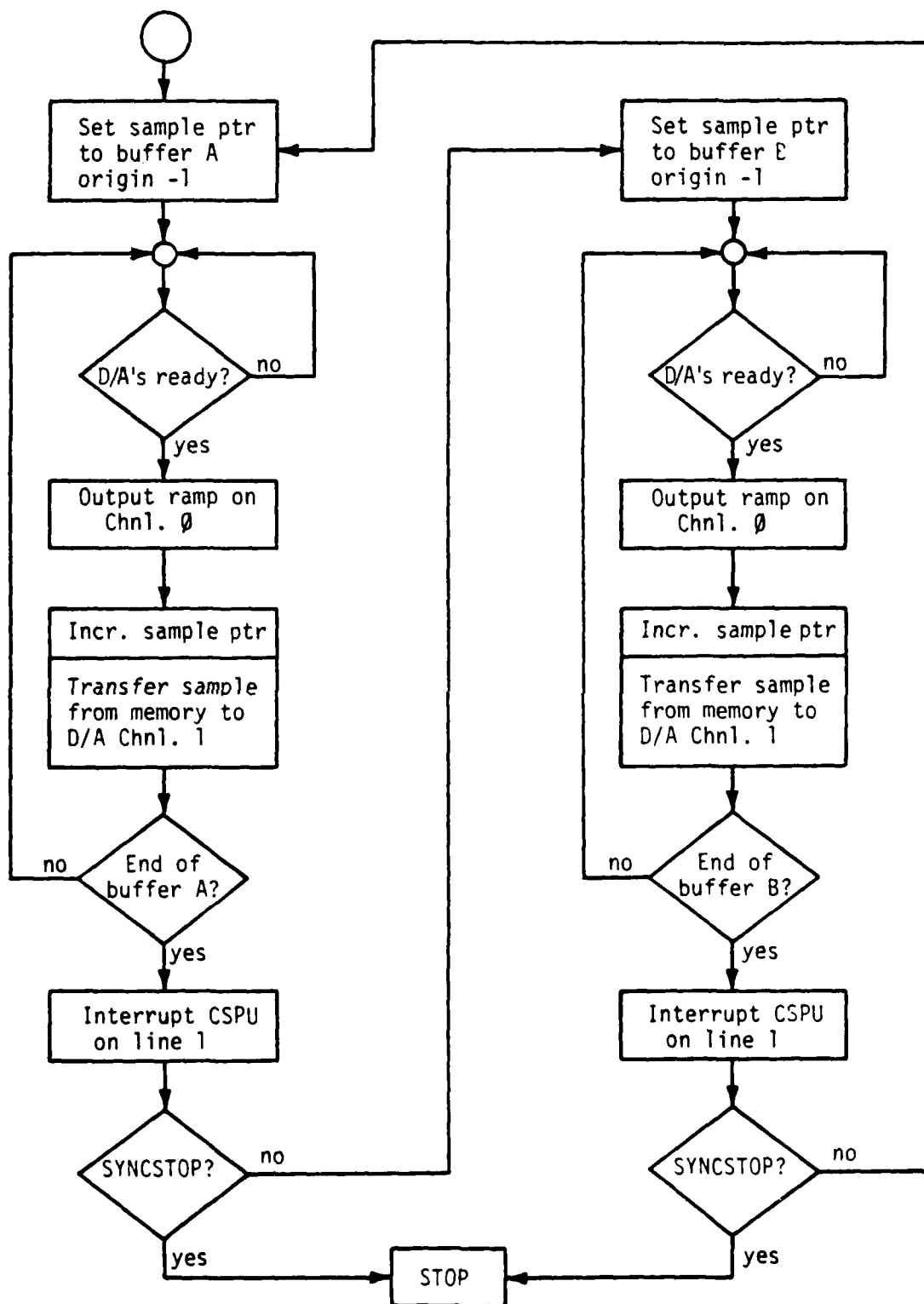


FIG. 15. DAPROG: AOM SCROLL PROGRAM

and set the flag RBTFA=1. The background process will execute SYNZB first, since RBTFA=1 and RSNFB is initialized to 0. SYNZB writes the first synthesized speech data in buffer RSNKB and sets the flag RSNFB=1. The AOMINT interrupt routine is initialized to expect the first synthetic speech data in buffer RSNKB, so when it is next activated, that buffer is copied into a D/A buffer for output by the AOM scroll program.

Meanwhile, the AOM scroll program has started by reading data (initialized to silence) from buffer RDABA, sending an interrupt, going on to RDABB, etc. These AOM interrupts activate AOMINT, whose task is to copy new synthesized speech data from RSINK buffers (initially RSNKB) to the just-emptied D/A buffers (initially RDABA). On its first several (usually 13) activations, AOMINT finds RSNKB empty, because the receiver has not yet gained frame-sync, so AOMINT copies a buffer of silence from buffer RSNKC to the D/A buffers instead. On the next activation after SYNZB has run, AOMINT finds RSNKB full, so it copies it to the D/A buffer, and thereafter all data buffers are alternately filled and emptied without conflict.

2.3 SYSTEM HARDWARE

The 16 kb/s speech coder system is implemented on a MAP-300 array processing computer, which is manufactured by CSP Inc., of Billerica, Mass. Section 2.3.1 describes the configuration of

MAP-300 equipment that is necessary for the speech coder system. Section 2.3.2 describes the additional audio signal interface and IOS-2SM scroll modifications for connection to a modem that complete the system.

2.3.1 MAP-300 Hardware

The MAP-300 configuration that was specified by Defense Communications Agency for the implementation of the speech coder system is listed below.

- 1 1030 MAP-300 Processor
- 1 2030 8K x 32 MOS Master Memory, 500 nsec, Bus 1
- 1 2050 16K x 32 MOS Slave Memory, 500 nsec, Bus 1
- 1 2203 8K x 32 MOS Master Memory, 300 nsec, Bus 2
- 1 2410 4K x 32 MOS Master Memory, 170 nsec, Bus 3
- 1 3110 PDP-11 Interface
- 1 4020 Model 2SM I/O Scroll
- 2 4040 Bus Switch (for Model 2SM I/O Scroll)
- 1 5120 Analog Data Acquisition Module
- 1 5130 Analog Output Module
- 1 6100 Expansion Chassis
- 1 6200 Auxiliary Power Supply

2.3.2 Audio and Modem Interface Hardware

In addition to the MAP-300 equipment listed above, two other pieces of equipment enable the MAP-300 to function as a complete, stand-alone speech coder system: an audio signal interface and a digital data interface to a modem. These two items were designed and built by the GTE Sylvania Electronic Systems Group in Needham, Mass. (see [1], Appendix A). Identical interfaces were provided for all three DCA MAP-300 systems so that they would be interchangeable at the hardware level.

The audio signal interface consists of a handset, tape input and output jacks, and circuitry for the amplification, equalization, and filtering necessary for interfacing speech input and output signals to the MAP-300 A/D and D/A converters.

The modem interface consists of modifications to the MAP-300 IOS-2SM scroll processor for the purpose of transferring data from MAP memory to the modem and also data from the modem into MAP memory. Two real-time clocks derived from a single master oscillator are also provided for controlling the modem data rate and the speech sampling rates.

2.4 SYSTEM SOFTWARE

The speech coder software consists of two distinct sets of modules. The first set is made up of modules that run in the MAP-300, including CSPI-supplied as well as BBN-written programs. Section 2.4.1 describes the MAP-300 modules that must be loaded into the MAP-300 processor before the speech coder can be operated.

The second set of software modules consists of FORTRAN routines that run in the PDP-11. These routines make use of the CSPI-supplied SNAP-II software system to initialize and start the MAP-300 speech coder programs. Section 2.4.2 describes these routines.

2.4.1 MAP-300 Software Components

All MAP-300 processing is done in conjunction with Release 3.5 of the CSPI-supplied SNAP-II software system. For the most part, BBN-written MAP-300 programs take the form of new AP or CSPU functions, callable via the standard SNAP-II calling procedure. In addition, BBN-written interrupt service routines and input/output scroll programs have been added to the executive.

CSPI-supplied MAP-300 software necessary for speech coder operation is described in Section 2.4.1.1. MAP-300 speech coder software written by BBN is described in Section 2.4.1.2.

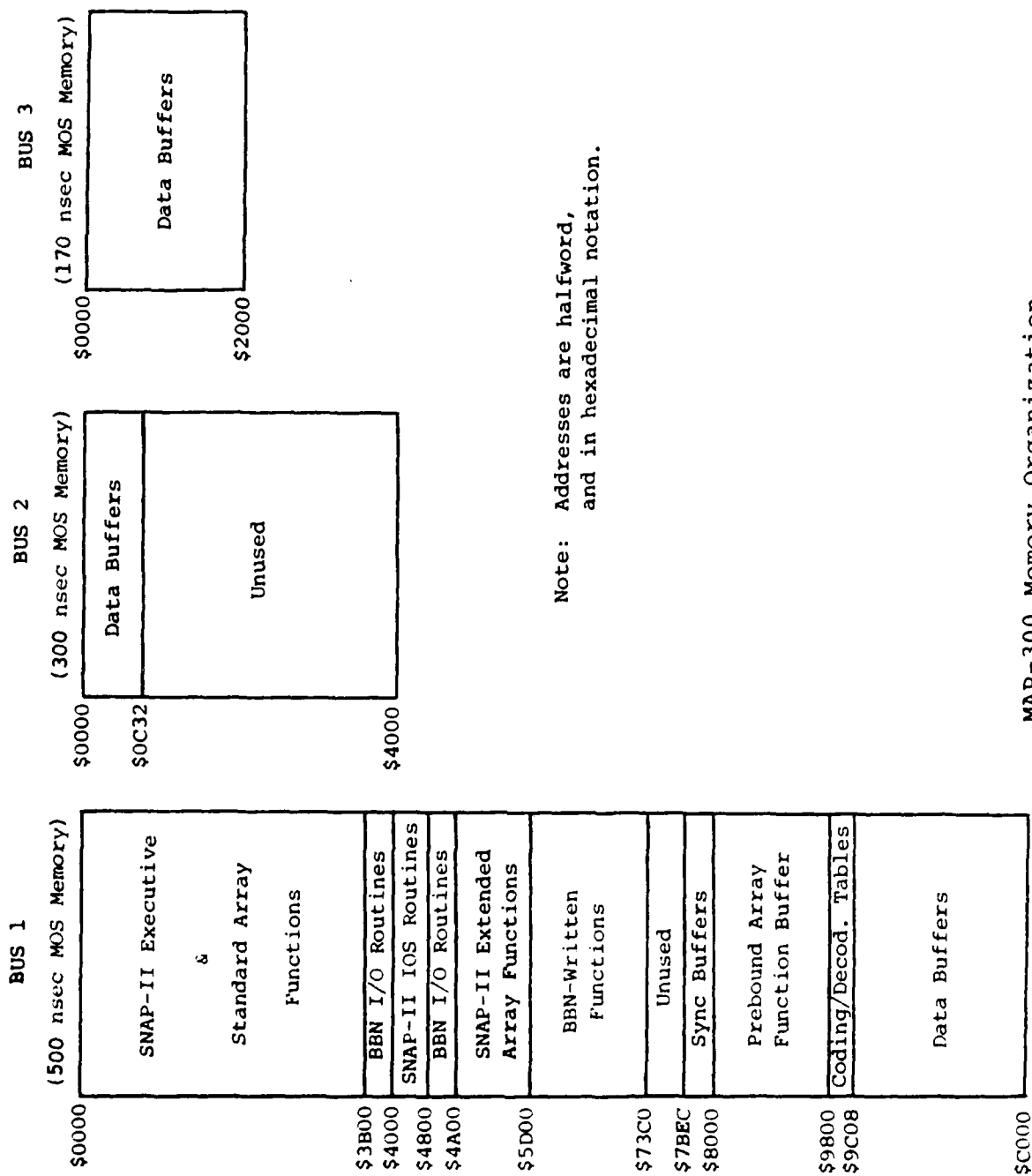
Fig. 16 shows the MAP-300 memory organization, including the location of CSPI and BBN software, as well as the location of buffers defined in the speech coder system.

2.4.1.1 CSPI-Supplied MAP-300 Software

The following CSPI-supplied MAP-300 software modules are required for speech coder operation:

SNAP-II Software System Release 3.5 Model 8300-RSX11M.
(This package includes the SNAP-II Executive and the
Standard and Extended Array Functions.)

SNAP-II Input/Output Scroll Package Release 0.1
Model 8400-RSX11M.
(This package includes the SNAP-II IOS Modules.)



MAP-300 Memory Organization
Fig. 16

These software modules are described in CSPI documentation [2,3,5].

2.4.1.2 BBN-written MAP-300 Software

BBN-written additions and modifications to the SNAP-II software system are contained in four separate files. (For all BBN-written MAP-300 modules, a file extension of .MSO designates "MAP source", .MOB designates "MAP object", and .MLI designates "MAP listing".) The files with first name "BBN16M" contain added SNAP-II functions (array and non-array) for algorithmic and system support processing. "BBN16T" contains tables for quantization and decoding. "BBN16U" contains the error protection and correction modules, ADAM, AOM, and IOS-2 programs, and CSPU routines that respond to interrupts from these devices. These programs and routines are described in Section 2.2 of this report. "BBN16P" contains patches to Release 3.5 of the SNAP-II executive. Assembly-listings of these four files appear in Appendix D of this report.

BBN-written SNAP-II functions are functionally described in Appendix A of this report. Each function has been assigned a Function Control Block (FCB) number. An entry has been made for each new FCB in the Function Dispatch Table (FDT), indicating the location of the APU, APS and/or CSPU module(s) that implement the associated function.

Several unused CSPI-supplied functions, normally available

with Release 3.5 of the SNAP-II software system, have been disabled to provide room in the FDT for BBN-written functions. The following SNAP-II functions are unavailable once the BBN-written MAP-300 software has been loaded.

CXMUL	(FCB# 182)	CMML	(FCB# 220)
CXMUL	(FCB# 183)	CMINV	(FCB# 224)
CSMAI	(FCB# 188)	MWLD	(FCB# 225)
CSMA2	(FCB# 189)	ADMRB	(FCB# 229)
FF2D	(FCB# 212)	VHIST	(FCB# 230)
		VRAN1	(FCB# 235)

These functions can be made available by reloading the SNAP-II software system without loading BBN-written MAP-300 software.

In addition, the standard SNAP-II interrupt routines for interrupts from Device 16 (Lines 1 and 2), Device 22 (Line 1), and Device 23 (Line 1) have been modified to transfer control to BBN-written interrupt routines. The standard interrupt routines for these devices can be similarly accessed by reloading the SNAP-II Executive without loading BBN-written MAP-300 software.

2.4.2 PDP-11 Software Components

The proper operation of the speech coder is defined and controlled by a group of programs running in the PDP-11. These programs include a set of SNAP-II support subroutines and a MAP-300 driver, supplied by CSPI, as well as a number of FORTRAN programs written by BBN that make use of the CSPI-supplied routines to perform the specific task of defining and controlling the operation of the speech coder in the MAP.

CSPI-supplied PDP-11 programs are described in Section 2.4.2.1. PDP-11 programs written by BBN are described in Section 2.4.2.2.

2.4.2.1 CSPI-Supplied PDP-11 Software

The following CSPI-supplied PDP-11 software modules are required for speech coder operation:

SNAP-II Software System Release 3.5 Model 8300-RSX11M.
(This package includes the SNAP-II Host Support Packages for Standard and Extended Array Functions.)

SNAP-II Input/Output Scroll Package Release 0.1 Model 8400-RSX11M
(This package includes the SNAP-II IOS Host Support Package.)

DEC RSX-11M I/O Driver Model 8901

These software modules are described in CSPI documentation [2,3,5,6,7].

2.4.2.2 BBN-written PDP-11 Software

A set of BBN-written FORTRAN programs performs the multiple tasks of defining and initializing MAP-300 buffers and scalars and of defining and executing the sequence of functions in the MAP that perform the actual speech coding operations. These FORTRAN programs are organized around a single mainline program (BBN16R), which calls, in turn, subroutines to configure MAP buffers and scalars (BBN16C), initialize these buffers and scalars (BBN16I), define function lists for various speech coder

tasks (BBN16F), and interact with the user in controlling the execution of these function lists (DCA96E). These subroutines are described in Sections 2.4.2.2.1 through 2.4.2.2.4 below.

2.4.2.2.1 Buffer and Scalar Configuration (BBN16C)

The task of defining MAP-300 buffers and scalars to the SNAP-II executive is performed by subroutine BBN16C. Upon entry, this routine first initializes the SNAP-II executive via the MPOPN function. All SNAP-II buffers are then configured using the MPCLB function, with buffer ID numbers, sizes, and addresses defined symbolically within the BBN16C routine. In general, transmitter buffer names begin with "T", while receiver buffer names begin with "R". Buffer sizes are stored in variables named by prefixing an "S" to the the buffer name. Similarly, buffer addresses are stored in variables named by prefixing "A" to the buffer name. The buffers used in the speech coder system are listed in Appendix B. All real scalars and integer scalars are then defined by assigning scalar ID numbers to the associated symbolic scalar names. Transmitter scalar names generally begin with "T", while receiver scalar names begin with "R". Real and integer scalars used in the speech coder system are listed in Appendix C.

All buffer and scalar names (and certain buffer sizes and addresses) are included in FORTRAN labeled COMMON blocks, which permit all other FORTRAN subroutines in the speech coder system

to reference these symbolic buffer and scalar names in their calls to SNAP-II functions.

2.4.2.2.2 Buffer and Scalar Initialization (BBN16I)

MAP-300 buffers and scalars are set to their initial values by subroutine BBN16I. Only certain buffers and scalars require such initialization. Appendices B and C indicate the proper initial contents of MAP buffers, real scalars, and integer scalars.

File "BBN16I" also contains several FORTRAN subroutines called by DCA96I and used to calculate the initial contents of certain buffers. Included are subroutines that generate a square-wave of a given frequency and amplitude (SQWAV) and a Hamming window of a given size (HAMMNG).

2.4.2.2.3 Control Structure Definition (BBN16F)

Subroutine BBN16F defines several function lists that specify the operation of the speech coder system. Function list ID numbers are defined symbolically. The FORTRAN variables containing the function list ID numbers are included in a labeled COMMON block so that they can be referenced by other subroutines.

As part of the function list definition task, subroutine BBN16F produces pre-bound versions of all of the SNAP-II array functions used in the speech coder system. Pre-binding is a SNAP-II operation that causes function parameter information,

normally communicated to the function at execution time, to be "bound" to the function at some earlier time (in this case at system initialization time). Approximately 6100 half-words are required on Bus 1 to store the pre-bound versions of the speech coder array functions.

Function lists ANLZA, ANLZB, SYNZA, and SYNZB are defined for the analysis and synthesis sections of the speech coder. (Since the system is double-buffered, the analysis and synthesis function lists are each defined twice to allow for the different sets of input and output buffers.) These function lists specify the sequence of MAP-300 array and non-array functions, operating on the system buffers and scalars previously defined, that implement the analysis and synthesis processes described in Sections 2.2.1.3 and 2.2.2.3 of this report.

The analysis and synthesis function lists described above are used in the specifications of several other function lists that define the outer structure of the speech coder system. Function list BBNLP defines the real-time speech coder loop, executing function lists ANLZA, SYNZA, ANLZB, and SYNZB when buffer status flags indicate that the input buffers are full and the output buffers are empty for each of these processes. Function list BBNRTS defines the start-up sequence for the real-time speech coder system, starting the Modem, ADAM, and AOM Scroll processors, and repeatedly executing the BBNLP function

list described above. This repeated execution is conditioned on the contents of MAP integer scalar RUN being non-zero; hence, the speech coder can be stopped by setting RUN to zero. Function list BBNRST restarts the real-time speech coder by setting RUN to non-zero and reinitiating the repeated execution of the real-time speech coder loop.

Two other function list definitions are included in this subroutine to support non-real-time file-to-file speech coder operation, speech coder timing operation, and speech coder oscilloscope display operation (for detailed internal timing). These modes of speech coder operation are not supported, and can not be invoked, in the delivered speech coder system. They were used for system development and debugging use under the RT-11 operating system and are included here to aid in future additions to the speech coder system. Function list BBNFFT specifies the speech coder loop with explicit calls to functions that execute the A/D, RMODEM, TMODEM, and D/A interrupt routines. Function list BBNTIM repeatedly invokes a sequence of six BBNFFT function list executions (one for each possible combination of input and output buffers), with software simulation of the transmitter-to-receiver communication path occurring after each BBNFFT execution.

All SNAP-II functions are called via PDP-11 support subroutines that perform the actual communication with the MAP

driver. These subroutines are contained in library SNPLIB for CSPI-supplied functions and in file BBN16H for BBN-written functions.

2.4.2.2.4 System Software Execution (BBN16E)

The execution of the speech coder system is controlled by subroutine BBN16E. This subroutine starts the real-time speech coder system by first loading the Modem, ADAM, and AOM Scrolls with their respective programs, and then executing the real-time speech coder start-up function list (BBNRTS) described in the preceding section. It then interacts with the user, responding to single-character commands to halt the speech coder ('Q'), enable/disable error simulation ('E'), enable/disable error correction ('C'), cause the speech coder to lose sync artificially ('L'), type out a group of speech coder state scalars ('T'), and suspend the controlling RSX-11M task, allowing the speech coder to continue executing in the MAP-300 ('S').

Provision is included in this subroutine for the user to select other speech coder modes of operation, specifically file-to-file, timing, and oscilloscope display modes. However, this mode selection process is bypassed in the delivered speech coder system, and real-time speech coder execution mode is forced. As described in the previous section, these other operating modes are not supported, and cannot be invoked, in the delivered speech coder system. They were used for system development and

debugging and are included here to aid in future additions to the speech coder system. File BBN16D contains dummy versions of various system-dependent subroutines, which constitute the unsupported software portions of these unsupported modes of speech coder operation.

2.5 SYSTEM TIMING PERFORMANCE

The speech coder system introduces a total delay of 9 frames, corresponding to 293.6 milliseconds, between voice input and synthesized voice output (not including any delays in transmission). This consists of 4 frames of delay introduced by the transmitter portion of the system and 5 frames of delay contributed by the receiver portion. In the transmitter, a frame of delay each is produced by the buffering in the ADAM scroll program, the ANALYZE processing module, and the TMODEM scroll program. In addition, the ANALYZE module introduces a frame of "folding" delay due to the concurrent execution of the ANALYZE module and the PROPAR/PRORES modules (see Section 2.2.1).

In the receiver, two frames of delay are contributed by the buffering in the RMODEM scroll program, and a frame each is produced by buffering in the SYNTHESIZE processing module and the AOM scroll program. In addition, the SYNTHESIZE module introduces a frame of "folding" delay due to the concurrent execution of the SYNTHESIZE module and the CORPAR and DECODA/B modules (see Section 2.2.2).

The observed internal timing of the speech coder modules is shown in Fig. 17. This figure reflects data obtained by running the speech coder in a loop without the ADAM, AOM, and MODEM scroll processors and observing the states of the MAP-300 RA flag (APU run) and G-flags with an oscilloscope. The speech coder was provided with simulated A/D input data. ADAM, AOM, and MODEM scroll interrupt routines were omitted from the timing loop. These interrupts were timed separately, and their execution times are indicated at the end of the figure. Including these interrupt routines, Fig. 17 shows a total processing time of 30.0 milliseconds for a 32.625 millisecond frame of data, or about 0.92 times real time. (This is the worst case total processing time. If the ADAM, AOM, or MODEM scroll interrupts were to occur while the AP were running and the CSPU were idle, some or all of the interrupt routine execution time would be hidden behind the concurrent AP routine.)

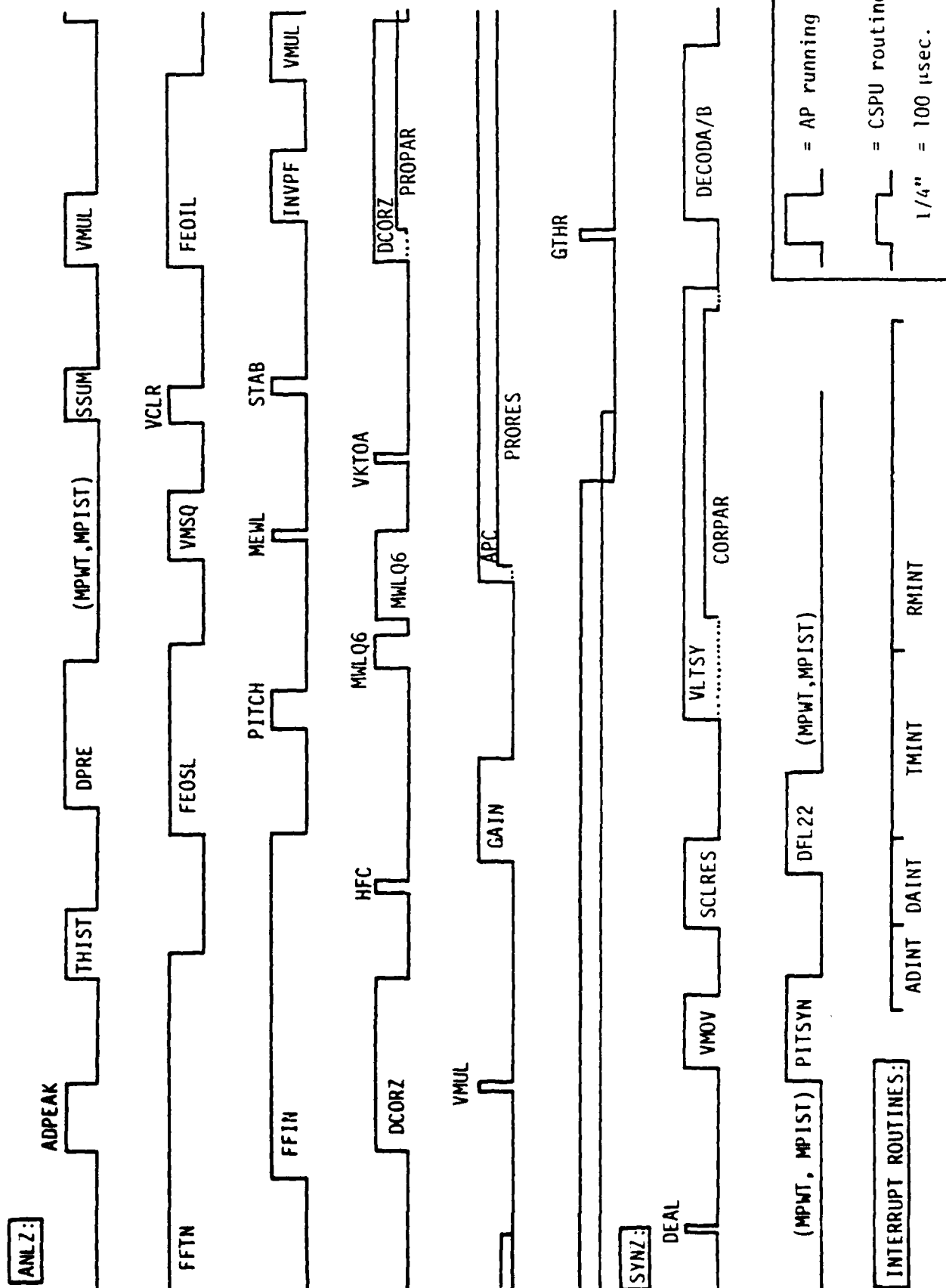


Figure 17 MAP-300 Timing

Bolt Beranek and Newman Inc.

Report No. 4565

3. SOFTWARE OPERATING PROCEDURES

The installation and execution of the speech coder system software are accomplished using the RSX-11M operating system as well as MAP-300 support software supplied by CSPI. It is assumed that the RSX-11M operating system, the MAP-300 device drivers, the MAP-300 loaders, and the SNAP-II software system have all previously been installed prior to the speech coder system installation.

After the speech coder system software has been installed, the speech coder can be executed by using the procedure given in Section 3.1. The software installation procedure is given in Section 3.2.

3.1 SOFTWARE EXECUTION PROCEDURE

The speech coder system execution procedure consists of two major steps. First, both MAP-300 processors in the full-duplex speech coder system must be loaded from the host PDP-11 with identical copies of the speech coder MAP-300 software. Second, two RSX-11M tasks must be run to initialize and start the two MAP-300 speech coders.

The following files must be disk-resident before speech coder execution can be attempted:

BBN16X.MBN

(MAP Binary file)

BBA16.TSK (Host Task for MAP #A)
BBB16.TSK (Host Task for MAP #B)

If these files do not exist, they must be created according to the speech coder system software installation procedure given in Section 3.2.

The following dialog indicates the proper command sequence for full-duplex speech coder system execution. User commands are underscored to differentiate them from computer responses.

```
>SET /UIC=[6,260]
>ALLOCATE MA:                    **allocate MAP #A**
>ALLOCATE MB:                    **allocate MAP #B**
>INSTALL BBA16
>INSTALL BBB16
>RUN [6,100]QLA                  **load MAP #A**
*BBN16X.MBN
TT1 -- STOP

>RUN [6,100]QLB                  **load MAP #B**
*BBN16X.MBN
TT1 -- STOP

>RUN BBA16                        **start MAP #A**
BBN 16K BPS SPEECH CODING SYSTEM

CONFIGURING MAP BUFFERS AND SCALARS...
INITIALIZING MAP BUFFERS AND SCALARS...
PREBINDING MAP FUNCTIONS AND DEFINING FUNCTION LISTS...
EXECUTING BBN16 SPEECH CODER...
SPEECH CODER IS IN OPERATION.

COMMANDS ARE:
S:    SUSPEND TASK (LEAVING SPEECH CODER RUNNING)
Q:    QUIT (HALT SPEECH CODER)
E N:  SIMULATE N ERRORS PER FRAME
C N:  ERROR-CORRECT IF N=0
L:    CAUSE RCVR TO LOSE SYNC
T:    TYPE OUT SPEECH CODER STATE

ENTER COMMAND: S
BBA16 -- PAUSE (SPEECH CODER CONTINUING)
```

```
>RUN BBB16                **start MAP #B**  
BBN 16K BPS SPEECH CODING SYSTEM
```

```
CONFIGURING MAP BUFFERS AND SCALARS...  
INITIALIZING MAP BUFFERS AND SCALARS...  
PREBINDING MAP FUNCTIONS AND DEFINING FUNCTION LISTS...  
EXECUTING BBN16 SPEECH CODER.....  
SPEECH CODER IS IN OPERATION.
```

COMMANDS ARE:

```
S:  SUSPEND TASK (LEAVING SPEECH CODER RUNNING)  
Q:  QUIT (HALT SPEECH CODER)  
E N: SIMULATE N ERRORS PER FRAME  
C N: ERROR-CORRECT IF N=0  
L:  CAUSE RCVR TO LOSE SYNC  
T:  TYPE OUT SPEECH CODER STATE
```

```
ENTER COMMAND: S  
BBB16 -- PAUSE (SPEECH CODER CONTINUING)
```

>

At this point in the command sequence, the full-duplex speech coder system should be in operation. Host tasks BBA16 and BBB16 can be resumed (for interaction with the speech coder software) by invoking the RSX-11M "RESUME" command. The speech coder system can be halted either by responding with "Q" to the "ENTER COMMAND:" typeout from BBA16 and BBB16 or by aborting the BBA16 and BBB16 tasks through the use of the RSX-11M "ABORT" command.

3.1.1 Optional Software Execution Procedure

Throughout the BBN16 Speech Coder MAP software, we have included code to set and clear various general-purpose flags (G-flags). These flags are used mostly for debugging, timing, and system measurement purposes; the manipulations are not required,

in general, for speech coder operation. Although these G-flag manipulations are strictly legal and proper according to CSPI documentation, they seem to induce "Bus 1 Memory Timeout" errors after apparently random amounts of time (ranging from a few minutes to several hours). For this reason, we have delivered with the system a patch file (NOGF16.MOB) which, when loaded into the MAP over the BBN16 speech coder software, no-op's all G-flag manipulations not needed for speech coder operation. This file can be loaded into the MAP processors (following the load of BBN16X) using the following command sequence:

```
>RUN [6,100]MALD
OBJECT INPUT? NOGF16.MOB
BINARY OUTPUT? (carriage return)
TT1 -- STOP 2
>RUN [6,100]MBLD
OBJECT INPUT? NOGF16.MOB
BINARY OUTPUT? (carriage return)
TT1 -- STOP 2
>
```

3.1.2 Analog Input Level Setting

The BBN16 speech coder software includes a facility for monitoring the peak input speech level. This facility allows the level to be set properly when the speech coder is being fed from an adjustable level signal source (such as a tape deck). The peak input level is maintained in integer scalar TADPK. TADPK can be displayed (along with a number of other scalars) during speech coder operation by entering a 'T' in response to the "COMMAND:" prompt. The peak input level range is 0-2047,

so the desired peak input level is approximately 1500-2000. A peak level above about 2000 introduces a potential input signal clipping condition, while a peak level below about 1500 does not take full advantage of the system's available dynamic range. The 'T' command resets TADPK to 0, so the displayed value pertains only to the time interval since the previous 'T' was typed.

3.2 SOFTWARE INSTALLATION PROCEDURE

The speech coder system software installation procedure consists of two major steps. First, the speech coder MAP-300 software must be converted from MAP object format to MAP binary format. Second, two RSX-11M tasks (for initializing and starting the two MAP-300 processors) must be task-built.

The following files must be disk-resident before speech coder software installation can be attempted:

BBN16X.MOB	(MAP object file)
	(This file consists of the following
	concatenated files, with intervening
	"FFFF" lines deleted:
	S300EX.MOB
	EAF300.MOB
	IOS300.MOB
	BBN16M.MOB
	BBN16U.MOB
	BBN16T.MOB
	BBN16P.MOB)
BBN16C.FOR	(HOST FORTRAN module)
BBN16D.FOR	(")
BBN16E.FOR	(")
BBN16F.FOR	(")
BBN16H.FOR	(")
BBN16I.FOR	(")

BBN16R.FOR (")

If these files do not exist, they must be recopied to disk from the RSX PIP-format "BBN 16 kb/s speech coder s/w" magnetic tape delivered with the speech coder system.

The MAP-300 speech coder software should be converted from MAP object format to MAP binary format using the following command sequence:

```
>RUN [6,100]MALD
OBJECT INPUT?BBN16X.MOB
BINARY OUTPUT?BBN16X.MBN
LOAD MAP?(Y OR N) N
TT1 -- STOP 2
>
```

The two RSX-11M tasks (for initializing and starting the two MAP-300 processors) should be generated next. First, all FORTRAN speech coder modules listed above must be compiled. Then the two tasks must be task-built, with each task including the compiled FORTRAN object modules, the CSPI-supplied SNAP host support library modules, and the appropriate CSPI-supplied MAP device driver for MAP #A or MAP #B. The following command sequence will accomplish this procedure:

```
>F4P BBN16C=BBN16C.FOR/CO:35
>F4P BBN16D=BBN16D.FOR/CO:35
>F4P BBN16E=BBN16E.FOR/CO:35
>F4P BBN16F=BBN16F.FOR/CO:35
>F4P BBN16H=BBN16H.FOR/CO:35
>F4P BBN16I=BBN16I.FOR/CO:35
>F4P BBN16R=BBN16R.FOR/CO:35
>TKB **build host task for MAP #A**
```

```
TKB>BBA16/FP/CP=BBN16R,BBN16C,BBN16D
TKB>BBN16E,BBN16F,BBN16H,BBN16I
TKB>[6,100]SNALIB/LB
TKB>/
ENTER OPTIONS:
TKB>UNITS=11
TKB>ASG=TI:7
TKB>ASG=MA:10
TKB>//
>TKB                                **build host task for MAP #B**
TKB>BBB16/FP/CP=BBN16R,BBN16C,BBN16D
TKB>BBN16E,BBN16F,BBN16H,BBN16I
TKB>[6,100]SNBLIB/LB
TKB>/
ENTER OPTIONS:
TKB>UNITS=11
TKB>ASG=TI:7
TKB>ASG=MB:10
TKB>//
>
```

The speech coder system software is now installed. The speech coder can be executed by using the procedure given in Section 3.1.

Bolt Beranek and Newman Inc.

Report No. 4565

4. REFERENCES

1. R. Viswanathan, J. Wolf, L. Cosell, K. Field, A. Higgins and W. Russell, "Design and Real-time Implementation of a Baseband LPC Coder for Speech Transmission over 9600 BPS Noisy Channels," Final Report, BBN Report No. 4327, Vols. I and II, Bolt Beranek and Newman Inc., Cambridge, MA, February 1980.
2. "Simple Notation for Array Processing, Version II (SNAP-II) Reference Manual," CSPI Document No. JB6000-017-01, C.S.P. Inc., Billerica, MA, November 1978.
3. "Release Note: SNAP-II Release 3 Executive Features and Expanded Array Function Set," CSPI Document No. DW6000-021-04, C.S.P. Inc., Billerica, MA, July 1979.
4. "Release Note: SNAP-II Extended Array Function Library, II," CSPI Document No. RB6000-24-00, C.S.P. Inc., Billerica, MA, March 1980.
5. "SNAP-II Input/Output Scroll Package, User's Manual, CSPI Document No. JW8400-001-01, C.S.P. Inc., Billerica, MA, July 1979.
6. "Installation Procedure for Release 3 SNAP-II Software System on the DEC RSX-11M System," CSPI Document No. LL8901-004-03, C.S.P. Inc., Billerica, MA, July 1979.
7. "MAP Software Interface Description for the DEC RSX-11M System," CSPI Document No. ST8901-000-04, C.S.P. Inc., Billerica, MA, July 1979.

Bolt Beranek and Newman Inc.

Report No. 4565

APPENDIX A
BBN-WRITTEN FUNCTION DESCRIPTIONS

This Appendix contains a complete user-level description for each MAP-300 SNAP-II function written by BBN and used in the speech coder implementation. The descriptions are ordered alphabetically by function name.

Functions that were used in the speech coder implementation and are not described here were supplied by CSPI as part of the SNAP-II Software System (see Sections 2.4.1.1 and 2.4.2.1).

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: ADINT
NAME EXPANSION: Simulate A/D interrupt

FCB #: 124
ARRAY OR NON-ARRAY FUNCTION: Non-array

APU MODULE NAME: --
APS MODULE NAME: --
CSPU MODULE NAME: ADAMINT

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
None			

NUMBER OF OUTPUT SAMPLES: --
NOTES:

FUNCTION DESCRIPTION:

This SNAP callable function calls the ADAMINT A/D Interrupt Service module in exactly the same manner as it is used to respond to A/D interrupts, so it may be used to simulate such an interrupt. See Section 2.2.1.2.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: ADPEAK (U)
NAME EXPANSION: Monitor peak signal from the A/D

FCB #: 182
ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: ADPKU
APS MODULE NAME: ADPKA
CSPU MODULE NAME: --

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
Buffer U	1-63	Real	Input

NUMBER OF OUTPUT SAMPLES: 1, in integer scalar TADPK
NOTES: --

FUNCTION DESCRIPTION:

Maintains a running peak absolute value of the data in the input buffers given to it on successive calls. The running peak is maintained as an integer value (to facilitate monitoring the peak level) in the integer scalar TADPK. Since SNAP-II standard binding does not support binding an integer scalar to an array function, the address of TADPK is "hard bound" in the APS module source code.

The input buffer is assumed to contain signals in the range -1 to +1 from a 12-bit A/D converter. The integer value maintained in TADPK falls in the range 0-2047.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: APC(Y,A,U,V,W,R,S,T)
NAME EXPANSION: APC loop

FCB #: 199
ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: APCUS
APS MODULE NAME: APCSS
CSPU MODULE NAME: SBMSAPC

PARAMETER DEFINITIONS:

PARAMETER	RANGE	SAMPLE TYPE	COMMENTS
Buffer Y	1-63	Fixed	Output: Coded residual samples
Real Scalar A	0-191	Real	Input: Pitch lag
Buffer U	1-63	Real (YBS)	Input: Preemphasized speech
Buffer V	1-63	Real (15)	Input: Filter coeffs: 6 spectral, 3 pitch, 6 noiseshaping.
Buffer W	1-63	Real (6)	Input: Quantizer segment scale factors, alternating with their reciprocals.
Buffer R	1-63	Real (432)	Input/Output: Pitch Filter History: last frame, current frame.
Buffer S	1-63	Real (6)	Input/Output: Spectral Filter History
Buffer T	1-63	Real (6)	Input/Output: Noise Shaping Filter History

NUMBER OF OUTPUT SAMPLES: $YBS + RBS/2 + SBS + TBS$

NOTES: Uses SA to modify APS code.
Requires special CSPU binding.

FUNCTION DESCRIPTION:

Compute and code APC residual. See Section 2.7 of the Algorithm Specification (Volume I, Appendix A of this report) for complete description.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: CORPAR(I)
NAME EXPANSION: Correct parameters

FCB #: 123
ARRAY OR NON-ARRAY FUNCTION: Non-array

APU MODULE NAME: --
APS MODULE NAME: --
CSPU MODULE NAME: CORPAR

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
Literal I	-2,0	Integer	Specifies buffer pair

NUMBER OF OUTPUT SAMPLES: --
NOTES:

FUNCTION DESCRIPTION:

Unbitstreams, error-corrects, and decodes parameters from one of the RBITS buffers to one of the RSOURCE buffers. The tables used for decoding each parameter are given in the file BBN16T.MS0.

The argument I specifies which pair of RBITS/RSOURCE buffers is used. I=-2 means RBTB/RSRA, and I=0 means RBTB/RSRB.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: DAINI
NAME EXPANSION: Simulate D/A Interrupt

FCB #: 127
ARRAY OR NON-ARRAY FUNCTION: Non-array

APU MODULE NAME: --
APS MODULE NAME: --
CSPU MODULE NAME: AOMINT

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
None			

NUMBER OF OUTPUT SAMPLES: --
NOTES:

FUNCTION DESCRIPTION:

The SNAP-callable function calls the AOMINT D/A Interrupt Service module, so it may be used to simulate such an interrupt. See Section 2.2.2.5.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: DEAL(Y,A,U,B,V,W)

NAME EXPANSION: Deal received decoded data from single buffer

FCB #: 212

ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: DEALUS

APS MODULE NAME: DEALSS

CSPU MODULE NAME: --

PARAMETER DEFINITIONS:

PARAMETER	RANGE	SAMPLE TYPE	COMMENTS
Buffer Y	1-63	Real (230)	Input: Received decoded data buffer
Real Scalar A	0-191	Real	Output: Received $\sqrt{\text{Energy}}$
Buffer U	1-63	Real (3)	Output: Received $\sqrt{\text{Delta Gains}}$
Real Scalar B	0-191	Real	Output: Received Pitch Lag
Buffer V	1-63	Real (3)	Output: Received Pitch Predictor Coeffs
Buffer W	1-63	Real (6)	Output: Received Spectral Predictor Coeffs

NUMBER OF OUTPUT SAMPLES: 14

NOTES:

FUNCTION DESCRIPTION:

On function exit, Y(0 - 215) is expected to contain 216 received residual samples.

SA <= Y(216)
U(0-2) <= Y(217-219)
SB <= Y(220)
V(0-2) <= Y(221-223)
W(0-5) <= Y(224-229)

AD-A096 092

BOLT BERANEK AND NEWMAN INC CAMBRIDGE MA

F/B 17/2.1

DESIGN AND REAL-TIME IMPLEMENTATION OF A ROBUST APC CODER FOR S--ETC(U)

DEC 80 J J WOLF, K D FIELD, W H RUSSELL

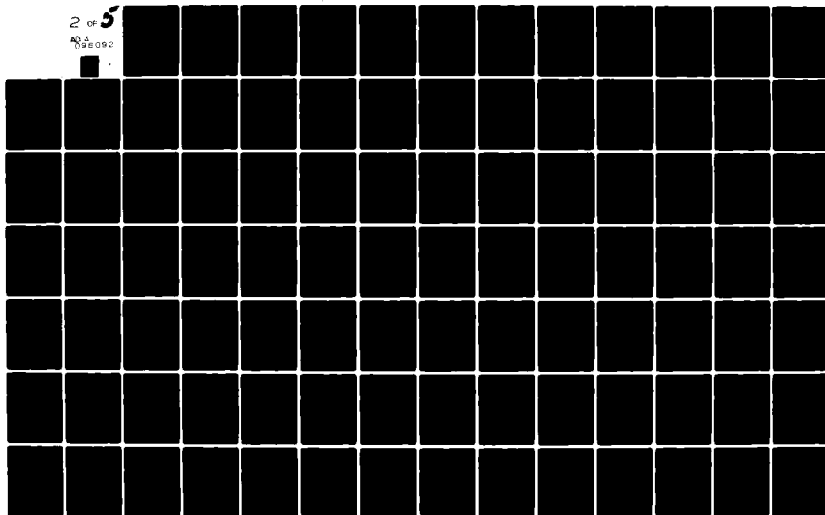
DCA100-79-C-0037

UNCLASSIFIED

BBN-4565-VOL-2

NL

2 of 5
NO. 096092



BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: DECODA
NAME EXPANSION: Decode residual, buffer-pair "A"

FCB #: 188
ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: DCDU
APS MODULE NAME: DCDSA
CSPU MODULE NAME: --

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
None			

NUMBER OF OUTPUT SAMPLES: --
NOTES:

FUNCTION DESCRIPTION:

Unbitstreams and decodes the residual samples from buffer RBTA to buffer RSRA. The table used for decoding the residual is given in the file BBN16T.MSO.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: DECODB
NAME EXPANSION: Decode residual, buffer-pair "B".

FCB #: 189
ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: DCDU
APS MODULE NAME: DCDSB
CSPU MODULE NAME: --

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
None			

NUMBER OF OUTPUT SAMPLES: --
NOTES:

FUNCTION DESCRIPTION:

Unbitstreams and decodes the residual samples from buffer RBTB to buffer RSRB. The table used for decoding the residual is given in the file 88N16T.M50.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: FFILR(Y,M,U,V,W)
NAME EXPANSION: Fast Fourier Transform, Not-in-Place, To Real Data, Long

PCB #: 198, 206

ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: --

APS MODULE NAME: -- (see below)

CSPU MODULE NAME: --

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>

(see below)

NUMBER OF OUTPUT SAMPLES: USB*2

NOTES: One complex sample is input beyond the end of the configured U buffer (see below).

FUNCTION DESCRIPTION:

This is a slightly modified version of the CSPI function FFILR (from SNAP-II, Extended Array Function Library II [4]). The modification involves an assumed reformatting of the complex frequency domain input (U) such that the last real input point is assumed to appear as such, rather than in the first imaginary input location (which is known to be zero).

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: FFTLR(Y,M,U,V,W)
NAME EXPANSION: Fast Fourier Transform, Not-in-Place, On Real Data, Long

FCB #: 204, 197

ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: --
APS MODULE NAME: -- (see below)
CSPU MODULE NAME: --

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>

(see below)

NUMBER OF OUTPUT SAMPLES: $UBS/2 + 1$

NOTES: One complex sample is output beyond the end of the configured Y buffer (see below).

FUNCTION DESCRIPTION:

This is a slightly modified version of the CSPI function FFTNR (from SNAP-II, Extended Array Function Library II [4]). The modification involves a reformatting of the complex frequency domain output (Y) such that the last real output point appears as such, rather than in the first imaginary output location (which is known to be zero).

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: GAIN(Y,A,U,B,V,C)
NAME EXPANSION: Compute Quantizer Segment Scale Factors

FCB #: 196
ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: GAINS
APS MODULE NAME: GAINSS
CSPU MODULE NAME: --

PARAMETER DEFINITIONS:

PARAMETER	RANGE	SAMPLE TYPE	COMMENTS
Buffer Y	1-63	Real (3)	Output: Quantizer segment scale factors, alternating with their reciprocals.
Real Scalar A	0-191	Real	Output: Coded frame energy
Buffer U	1-63	Real (3)	Output: Coded segment delta-gains
Real Scalar B	0-191	Real	Input: 1/72
Buffer V	1-63	Real (216)	Input: Second Residual Samples
Real Scalar C	0-191	Real	Input: 1/3

NUMBER OF OUTPUT SAMPLES: 7 total

NOTES: Fixed to process 216 input buffer V elements, in 3 segments of 72 elements each.

FUNCTION DESCRIPTION:

$$SG(n) = \frac{1}{72} \sum_{i=1}^{72} (V(i+(n*72)))^2 \quad n = 0,1,2$$

$$G = \frac{1}{3} \sum_{i=1}^{72} (SG(i)) ; GH = \text{quantized } \sqrt{G}; SA = \text{coded } G$$

$$DG(n) = SG(n)/(GH*GH) \quad n = 0,1,2$$

$$U(n) = \text{coded } DG(n) ; DGH(n) = \text{quantized } \sqrt{DG(n)} \quad n=0,1,2$$

$$\left. \begin{aligned} Y(2n) &= (GH*DGH(n)) \\ Y(2n+1) &= 1./Y(2n) \end{aligned} \right\} \quad n=0,1,2$$

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: GTHR(Y,A,U,B,V,W)
NAME EXPANSION: Gather Transmission data into single buffer

FCB #: 200

ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: GTHRUS

APS MODULE NAME: GTHRSS

CSPU MODULE NAME: --

PARAMETER DEFINITIONS:

PARAMETER	RANGE	SAMPLE TYPE	COMMENTS
Buffer Y	1-63	Fixed (230)	Output: Transmission buffer
Real Scalar A	0-191	Fixed (in lefthword)	Input: Coded Energy
Buffer U	1-63	Fixed (3)	Input: Coded Delta Gains
Real Scalar B	0-191	Fixed (in lefthword)	Input: Coded Pitch Lag
Buffer V	1-63	Fixed (3)	Input: Coded Pitch Predictor Coeffs.
Buffer W	1-63	Fixed (6)	Input: Coded Spectral Predictor Coeffs.

NUMBER OF OUTPUT SAMPLES: YBS-216

NOTES:

FUNCTION DESCRIPTION:

On function call, Y(0-215) is expected to contain 216 coded residual samples.

Y(216) <= SA
Y(217-219) <= U(0-2)
Y(220) <= SB
Y(221-223) <= V(0-2)
Y(224-229) <= W(0-5)

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: HFC(Y,U,V)
NAME EXPANSION: High Frequency Correction

FCB #: 190
ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: HFCUS
APS MODULE NAME: HFCSS
CSPU MODULE NAME: --

PARAMETER DEFINITIONS:

PARAMETER	RANGE	SAMPLE TYPE	COMMENTS
Buffer Y	1-63	Real	Output: HFC'd autocorrelation coeffs
Buffer U	1-63	Real	Input: Autocorrelation coeffs
Buffer V	1-63	Real (3)	Input: LAMBDA* $\mu(i)$ LAMBDA = .035 $\mu = \begin{pmatrix} .375 \\ -.25 \\ .0625 \end{pmatrix}$

NUMBER OF OUTPUT SAMPLES: YBS
NOTES:

FUNCTION DESCRIPTION:

For $i = 0, 2$:

$$Y(i) = U(i) + E2*V(i)$$

For $i = 2, (YBS-1)$:

$$Y(i) = U(i)$$

$$K1 = U(1)/U(0)$$

$$E1 = (1 - K1^2)*U(0)$$

$$K2 = (U(2) - K1*U(1))/E1$$

$$E2 = E1*(1 - K2^2)$$

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: INVPF(Y,A,U,V)
NAME EXPANSION: 3-tap Inverse Pitch Filter

FCB #: 167
ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: INVPU5
APS MODULE NAME: INVPSI
CSPU MODULE NAME: --

PARAMETER DEFINITIONS:

PARAMETER	RANGE	SAMPLE TYPE	COMMENTS
Buffer Y	1-63	Real	Output: filter output
Real Scalar A	50-191	Real	Input: pitch in sample-times
Buffer U	1-63	Real (YBS+216)	Input: filter input, including previous 216 samples.
Buffer V	1-63	Real (3)	Input: Filter coefficients

NUMBER OF OUTPUT SAMPLES: YBS

NOTES: This routine uses SA to modify its APS code. Buffer U must be compact 32-bit floating point.

FUNCTION DESCRIPTION:

Implements a 3-tap FIR pitch filter. Let M = SA, the pitch in terms of sample times:

$$y(n) = u(n) + \sum_{j=0}^2 v(j) * u(n - M - j + 1), \quad n=0,1,\dots,YBS-1$$

The pitch can take on values $14 \leq M \leq 133$, so the buffer U contains the "previous" frame of 216 samples in addition to the "current" frame. That is, $U(0) = u(-216)$, $U(216) = u(0)$, etc.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: MPGSC(GFLAG,SETCLR)
NAME EXPANSION: G-Flag Set/Clear

FCB #: 106
ARRAY OR NON-ARRAY FUNCTION: Non-array

APU MODULE NAME: -
APS MODULE NAME: -
CSPU MODULE NAME: MPGSC

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
Literal flag	0-3	Integer	Selects G-flag
Literal SETCLR	0-1	Integer	SETCLR=0=>Set flag SETCLR=1=>Clear flag

NUMBER OF OUTPUT SAMPLES: -
NOTES:

FUNCTION DESCRIPTION:

MPGSC sets or clears one of the four G-flags. This is useful for program timing using an external oscilloscope.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: MPIFF(IA,IB, FLID)

NAME EXPANSION: If (IA.NE.0) & (IB.EQ.0) Conditional Function List Execution

FCB #: 105

ARRAY OR NON-ARRAY FUNCTION: Non-array

APU MODULE NAME: -

APS MODULE NAME: -

CSPU MODULE NAME: MPIFFS

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
Integer Scalar IA	0-127	Integer	Input
Integer Scalar IB	0-127	Integer	Input
Literal FLID	0-63	Integer	Input: Function List ID

NUMBER OF OUTPUT SAMPLES: -

NOTES: Function list 'FLID' must be previously defined.

FUNCTION DESCRIPTION:

Function list 'FLID' is executed if and only if Integer Scalar IA is not equal to zero, and Integer Scalar IB is equal to zero.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: MWLQ6(Y,A,U,V,W)
NAME EXPANSION: Matrix (Weinter-Levinson-Durbin) Solution, with
 Quantized and coded outputs, 6th order
FCB #: 135
ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: MWLFSAPU
APS MODULE NAME: MWLFSAPS
CSPU MODULE NAME: MWLQSSSM

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
Buffer Y	1-63	Real (12)	Output: Reflection coefficients and error terms (as in MWLD)
Real Scalar A	0-191	Real	Input: Threshold value (as in MWLD)
Buffer U	1-63	Real (6)	Output: Quantized reflection coefficients
Buffer V	1-63	Real (7)	Input: Autocorrelation coefficients (as in MWLD).
Buffer W	1-63	Long Fixed (6)	Output: Coded reflection coefficients.

NUMBER OF OUTPUT SAMPLES: 6

NOTES: Special intermediate CSPU support is used.

FUNCTION DESCRIPTION:

This module consists of the SNAP-II function MWLD followed by a special quantization/coding module. MWLD outputs LPC coefficients into the buffer U, but the Q/C module overwrites them with the quantized reflection coefficients. The threshold and quantized-value tables are not specified as parameters, but are referenced symbolically by the APS module.

MWLD requires that buffers Y,U,V be compact (SI=1). Buffer W must be compact also.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: PITCH(Y,A,U,B,C,D)
NAME EXPANSION: Compute and code pitch

FCB #: 134
ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: PITUS
APS MODULE NAME: PITSS
CSPU MODULE NAME: --

PARAMETER DEFINITIONS:

PARAMETER	RANGE	SAMPLE TYPE	COMMENTS
Buffer Y	1-63	Real (3)	Output: Autocorrelation coefficients -RP(Pitch lag -1) -RP(Pitch lag) -RP(Pitch lag +1)
Real Scalar A	0-191	Real	Output: Pitch lag (in samples)
Buffer U	1-63	Real (512)	Input: Autocorrelation coefficients
Real Scalar B	0-191	Fixed (in lefthword)	Output: Coded pitch
Real Scalar C	0-191	Real	Input: Constant 14.
Real Scalar D	0-191	Real	Output: Single tap coefficient

NUMBER OF OUTPUT SAMPLES: 3 Output Scalars, 3 Output Buffer samples

NOTES: Uses modified SMAX.
Only uses part of input buffer U.
Uses SA to modify APS code

FUNCTION DESCRIPTION:

A = index of $\max_{i=14}^{133} [U(i)]$; where first U element is U(0).

Left half-word of B = A - 14. = A - C

(This value is generated by multiplying by 2**-15, and ALIGN'ing.)

$D = \frac{-U(\text{pitch lag})}{U(0)}$ (single tap coefficient (for case of 3-taps unstable))

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: PITSYN(Y,A,U,V)
NAME EXPANSION: 3-tap direct form pitch-synthesis recursive filter

FCB #: 203
ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: PITSUS
APS MODULE NAME: PITSSI
CSPU MODULE NAME: --

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u> (length)	<u>COMMENTS</u>
Buffer Y	1-63	Real (UBS+216)	Output: filter output plus 216 preceding output samples.
Real Scalar A	50-191	Real	Input: Pitch in samples
Buffer U	1-63	Real	Input: Filter input
Buffer V	1-63	Real (3)	Input: Filter Coefficients

NUMBER OF OUTPUT SAMPLES: UBS

NOTES: Uses value of SA to modify the APS code. Buffer Y must be compact 32-bit floating point.

FUNCTION DESCRIPTION:

Implements a 3-pole recursive filter for pitch synthesis. Let M=SA, the pitch expressed in samples

$$y(n) = u(n) - \sum_{j=0}^2 v(j) * y(n-M-j+1), \quad n=0,1,\dots,UBS-1$$

The pitch can take on values $14 \leq M \leq 133$, so the buffer Y must contain "previous" samples; hence, it functions as both input (previous output) and output (output computed by the function). For buffer-configuration reasons, Y must contain the previous 216 samples of output.

Buffer Y is defined so that the first value in it corresponds to y(-216).

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: PROPAR(I)
NAME EXPANSION: Protect analysis parameters

FCB #: 122
ARRAY OR NON-ARRAY FUNCTION: Non-array

APU MODULE NAME: --
APS MODULE NAME: --
CSPU MODULE NAME: PROPAR

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
Literal I	-2,0	Integer	Specifies buffer pair

NUMBER OF OUTPUT SAMPLES: --
NOTES:

FUNCTION DESCRIPTION:

Error protects and bitstreams analysis parameters from one of the TSINK buffers into one of the TBITS buffers; also accumulates histogram statistics for all the analysis parameters except pitch.

The argument I specifies which pair of TSINK/TBITS buffers is used. I=-2 means TSINKA/TBITA, and I=0 means TSINKB/TBITB.

See Section 2.2.1.3.18.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: PRORES(I)
NAME EXPANSION: "Protect" residual samples

FCB #: 121

ARRAY OR NON-ARRAY FUNCTION: Non-array

APU MODULE NAME: --

APS MODULE NAME: --

CSPU MODULE NAME: PRORES

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
Literal I	-2,0	Integer	Specifies buffer pair

NUMBER OF OUTPUT SAMPLES: --

NOTES:

FUNCTION DESCRIPTION:

Bitstreams coded residual samples from one of the TSINK buffers into one of the TBITS buffers. No "protection" is done.

The argument I specifies which pair of TSINK/TBITS buffers is used. I=-2 means TSNKA/TBTA, and I=0 means TSNKB/TBTB.

See Section 2.2.1.3.18.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: RMINT
NAME EXPANSION: Simulate Receiver Modem Interrupt

FCB #: 126
ARRAY OR NON-ARRAY FUNCTION: Non-array

APU MODULE NAME: --
APS MODULE NAME: --
CSPU MODULE NAME: RMODEMINT

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
None			

NUMBER OF OUTPUT SAMPLES: --
NOTES:

FUNCTION DESCRIPTION:

This SNAP-callable function calls the RMODEMINT Receiver Modem Interrupt Service module, so it may be used to simulate such an interrupt. See Section 2.2.2.2.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: SCLRES(Y,A,U,V)
NAME EXPANSION: Compute Quantizer scale factors and scale residual

FCB #: 202
ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: SCLS
APS MODULE NAME: SCLSS
CSPU MODULE NAME: --

PARAMETER DEFINITIONS:

PARAMETER	RANGE	SAMPLE TYPE	COMMENTS
Buffer Y	1-63	Real (216)	Output: Scaled residual samples
Real Scalar A	0-191	Real	Input: Received Frame Energy
Buffer U	1-63	Real (3)	Input: Received Delta Gains
Buffer V	1-63	Real	Input: Received Residual samples

NUMBER OF OUTPUT SAMPLES: 216
NOTES:

FUNCTION DESCRIPTION:

$GFAC(j) = [SA * U(j)] \quad j = 1, 2, 3$
 $Y(i) = V(i) * GFAC(j) \quad \begin{matrix} j = 1 \text{ for } i = 1, 72 \\ j = 2 \text{ for } i = 73, 144 \\ j = 3 \text{ for } i = 145, 216 \end{matrix}$

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: STAB(Y,A,U,V)
NAME EXPANSION: Stability check on pitch predictor coefficients; code and quantize.

FCB #: 150

ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: STABS

APS MODULE NAME: STABSS

CSPU MODULE NAME: --

PARAMETER DEFINITIONS:

PARAMETER	RANGE	SAMPLE TYPE	COMMENTS
Buffer Y	1-63	Real	Output: Quantized stable pitch predictor coeffs.
Real Scalar A	0-191	Real	Input: Single Tap coeff.
Buffer U	1-63	Fixed	Output: Coded stable pitch predictor coeffs.
Buffer V	1-63	Real	Input: Pitch Predictor coefficients.

NUMBER OF OUTPUT SAMPLES: 3 Quantized, 3 coded values

NOTES: Module fixed to process 3 pitch predictor coeffs.

FUNCTION DESCRIPTION:

Do stability check on $V(0)$, $V(1)$, $V(2)$:

$$T1 = V(0) + V(1) + V(2)$$

$$T2 = V(0) - 2V(1) + V(2)$$

$$T3 = V(0) - V(2)$$

If $[(T1 < -1) .OR. (T2 < -1) .OR. (T3 < -1) .OR. (T2 > +2) .OR. (T3 > +1)]$

then unstable: set $C1 = 0$
 $C2 = SA$
 $C3 = 0$

Else, stable: set $C1 = V(0)$
 $C2 = V(1)$
 $C3 = V(2)$

Code and quantize $C1$, $C2$, $C3$, set Y buffer to quantized values,
set U buffer to coded values.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: THIST(Y,U,V,W,R,S)
NAME EXPANSION: Set up transmitter frame histories.

FCB #: 201
ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: THUS
APS MODULE NAME: THSS
CSPU MODULE NAME: SBMSTH

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
Buffer Y	1-63	Real	Output: Preemphasized speech: history,current frame
Buffer U	1-63	Real (216)	Input: Preemphasized speech: current
Buffer V	1-63	Real	Output: First residual: history,current frame
Buffer W	1-63	Real (6)	Input: First residual: last 6 elements of current frame
Buffer R	1-63	Real	Output: APC pitch filter history last,current frame
Buffer S	1-63	Real (216)	Input: APC pitch filter history: current frame

NUMBER OF OUTPUT SAMPLES: UBS + WBS + SBS

NOTES: Requires special CSPU binding.

FUNCTION DESCRIPTION:

$Y(0) - Y(UBS - 1) \leq U(0) - U(UBS - 1)$
 $V(0) - V(WBS - 1) \leq W(0) - W(WBS - 1)$
 $R(0) - R(SBS - 1) \leq S(0) - S(SBS - 1)$

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: TMINT
NAME EXPANSION: Simulate Transmitter Modem Interrupt

FCB #: 125
ARRAY OR NON-ARRAY FUNCTION: Non-array

APU MODULE NAME: --
APS MODULE NAME: --
CSPU MODULE NAME: TMODEMINT

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
None			

NUMBER OF OUTPUT SAMPLES: --
NOTES:

FUNCTION DESCRIPTION:

This SNAP-callable function calls the TMODEMINT Transmitter Modem Interrupt Service module, so it may be used to simulate such an interrupt. See Section 2.2.1.5.

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: VKTOA(Y,U)
NAME EXPANSION: Convert Reflection Coefficients (K) to Linear Prediction Coefficients (A)

FCB #: 133

ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: VKAUS

APS MODULE NAME: VKASI

CSPU MODULE NAME: --

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
Buffer Y	1-63	Real	Output: LP coefficients A(1), A(2), ..., A(UBS)
Buffer U	1-63	Real	Input: Reflection Coefficients K(1), K(2), ..., K(UBS)

NUMBER OF OUTPUT SAMPLES: UBS

NOTES: This function is different from VKTOA in the 9.6 kb/s MAP-300 system; that was specific to an 8-pole system; this version is general. Also, this version does not output A(0).

FUNCTION DESCRIPTION:

The LPC coefficients $A(i)$, $i = 0, 1, \dots, p$ are computed from the reflection coefficients $K(j)$, $j = 1, 2, \dots, p$, by the recursion:

$$A_m(m) = K(m)$$

$$A_m(L) = A_{m-1}(L) + K(m)A_{m-1}(m-L) \quad \begin{matrix} L = 1, \dots, m-1 \\ m = 1, \dots, p \end{matrix}$$

where $A_m(L)$ indicates the value of the L-th coefficient on the m-th iteration.

The output buffer does not contain A(0), which is 1.0. The first output value is A(1).

BBN-WRITTEN SNAP-II FUNCTION DESCRIPTION

FUNCTION NAME: VLTSY(Y,U,V,W)
NAME EXPANSION: Lattice Synthesis Filter

FCB #: 132
ARRAY OR NON-ARRAY FUNCTION: Array

APU MODULE NAME: VLTSY\$
APS MODULE NAME: V3200\$
CSPU MODULE NAME: -

PARAMETER DEFINITIONS:

<u>PARAMETER</u>	<u>RANGE</u>	<u>SAMPLE TYPE</u>	<u>COMMENTS</u>
Buffer Y	1-63	Real	Output synthetic speech samples
Buffer U	1-63	Real(8)	Filter memory (G's)
Buffer V	1-63	Real(8)	Reflection coefficients (K's)
Buffer W	1-63	Real	Input samples

NUMBER OF OUTPUT SAMPLES: WBS

NOTES: Exactly 8 elements from each U and V are used. Contents of U are changed.

FUNCTION DESCRIPTION:

Performs lattice form all-pole filter, using reflection coefficients as filter coefficients.

Implements the following Fortran code:

```
DO 20 I=1, WBS
  F(7)=W(I)-G(7)*K(8)
  DO 10 J=6,0, -1
    F(J)=F(J+1)-G(J)*K(J+1)
    G(J+1)=G(J)+F(J)*K(J+1)
    G(0)=F(0)
  10 Y(I)=F(0)
20
```

APPENDIX B
MAP-300 BUFFERS

The table contained in this Appendix describes the characteristics of each data buffer used in the MAP-300 speech coder implementation. The table entry labeled "BID" indicates the buffer identification number (for SNAP-accessible buffers) or is blank (for non-SNAP buffers). The entries labeled "Written by:" and "Read by:" indicate subroutines, function lists, or program modules that so access each buffer. (The A/D, D/A, transmitter modem, and receiver modem interrupt service routines are specified in these tables by "ADINT", "DAINT", "TMINT", and "RMINT", respectively.)

MAP-300 Buffers

Usage key: 1 - constant
2 - variable

Name	BID	Description	Positional Characteristics	Written by:	Read by:	Sample Incr't	Sample Type	Number of Samples	Halfword Address	Bus #	Initial Values	Usage
CTHR1 } CTHR3 }	-	Coding Table of Pitch Tap Thresholds K(1, 3)	Fixed Address	--	ANLZ (Pitch)	1	RL LING	8	38912	1	(see BBN161)	1
CTHR2	--	Coding Table of Pitch Tap Thresholds K(2)	Fixed Address	--	ANLZ (Pitch)	1	RL LING	16	38928	1	(see BBN161)	1
CTHR1	--	Coding Table of K(1) Thresholds	Fixed Address	--	ANLZ (Spec)	1	RL LING	64	38960	1	(see BBN161)	1
CTHR2	--	Coding Table of K(2) Thresholds	Fixed Address	--	ANLZ (Spec)	1	RL LING	32	39088	1	(see BBN161)	1
CTHR3	--	Coding Table of K(3) Thresholds	Fixed Address	--	ANLZ (Spec)	1	RL LING	16	39152	1	(see BBN161)	1
CTHR4	--	Coding Table of K(4) Thresholds	Fixed Address	--	ANLZ (Spec)	1	RL LING	16	39184	1	(see BBN161)	1
CTHR5	--	Coding Table of K(5) Thresholds	Fixed Address	--	ANLZ (Spec)	1	RL LING	16	39216	1	(see BBN161)	1

MAP-300 Buffers

Usage key: 1 = constant
2 = variable

Name	BID	Description	Positional Characteristics	Written by:	Read by:	Sample Incr't	Sample Type	Number of Samples	Halfword Address	Bus #	Initial Values	Usage
CTHR6	--	Coding Table of K(6) Thresholds	Fixed Address	--	AMLZ (Spec)	1	RL LNG	16	39248	1	(see BBR16T)	1
CTHG	--	Coding Table of Energy Thresholds	Fixed Address	--	AMLZ (Gain)	1	RL LNG	64	39280	1	(see BBR16T)	1
CTHNG	--	Coding Table of Delta Gain Thresholds	Fixed Address	--	AMLZ (Gain)	1	RL LNG	4	39408	1	(see BBR16T)	1
CTHU	--	Coding Table of Residual Sample Thresholds	Fixed Address	--	AMLZ (APC)	1	RL LNG	4	39416	1	(see BBR16T)	1
DWL C1 DWL C3	--	Decoding Table of Pitch Tap Values C1, C3 (3bits)	Fixed Address	--	AMLZ (Pitch) CORPAR	1	RL LNG	8	39424	1	(see BBR16T)	1
DWL C2	--	Decoding Table of Pitch Tap Values C2 (4bits)	Fixed Address	--	AMLZ (Pitch) CORPAR	1	RL LNG	16	39440	1	(see BBR16T)	1
DWL K1	--	Decoding Table of Reflection Coefficient Values K(1) (6 bits)	Fixed Address	--	AMLZ (Spec) CORPAR	1	RL LNG	64	39472	1	(see BBR16T)	1

MAP-300 Buffers

Usage key: 1 - constant
2 - variable

Name	BID	Description	Positional Characteristics	Written by:	Read by:	Sample Increment	Sample Type	Number of Samples	Halfword Address	Bus #	Initial Values	Usage
DWL K2	--	Decoding Table of Reflection Coefficient Values K(2) (5 bits)	Fixed Address	--	ANL Z (Spec) CORPAR	1	RL LING	32	39600	1	(see BBN161)	1
DWL K3	--	Decoding Table of Reflection Coefficient Values K(3) (4 bits)	Fixed Address	--	ANL Z (Spec) CORPAR	1	RL LING	16	39664	1	(see BBN161)	1
DWL K4	--	Decoding Table of Reflection Coefficient Values K(4) (4 bits)	Fixed Address	--	ANL Z (Spec) CORPAR	1	RL LING	16	39696	1	(see BBN161)	1
DWL K5	--	Decoding Table of Reflection Coefficient Values K(5) (4 bits)	Fixed Address	--	ANL Z (Spec) CORPAR	1	RL LING	16	39728	1	(see BBN161)	1
DWL K6	--	Decoding Table of Reflection Coefficient Values K(6) (4 bits)	Fixed Address	--	ANL Z (Spec) CORPAR	1	RL LING	16	39760	1	(see BBN161)	1
DWL G	--	Decoding Table of Energy Value (6 bits)	Fixed Address	--	ANL Z (Gain) CORPAR	1	RL LING	64	39792	1	(see BBN161)	1
DWL DG	--	Decoding Table of Delta Gain Values (2 bits)	Fixed Address	--	ANL Z (Gain) CORPAR	1	RL LING	4	39920	1	(see BBN161)	1

MAP-300 Buffers

Usage key: 1 = constant
2 = variable

Name	BIT	Description	Positional Characteristics	Written by:	Read by:	Sample Incr't	Sample Type	Number of Samples	Halfword Address	Bus #	Initial Values	Usage
DVLU	--	Decoding Table of Residual Samples Values (2 bits)	Fixed Address	--	ANLZ (APC)	1	RL LNG	4	39928	1	(see BBN161)	1
DVLUF	--	Folded Decoding Table of Residual Samples	Fixed Address	--	DECODE/B	1	RL LNG	4	39936	1	(see BBN161)	1
HISTO	--	Histogram Data	Fixed Address	PROPAR	HIST16.SAV	1	RL LNG	160	39944	1	zeros	2
TADBA	--	A/D Input from ADAM	--	ADAM Scroll Program	ADINT	1	RL SHRT	216	40264	1	--	2
TADRB	--	A/D Input from ADAM	--	ADAM Scroll Program	ADINT	1	RL SHRT	216	40480	1	--	2
TADBC	--	A/D "On-hook" Tone Data	--	BBN161	ADINT	1	RL SHRT	216	40696	1	Approx. 240 Hz Square wave	1
TSRA	1	Input Speech to ANLZA	--	ADINT	ANLZ (Premp)	1	RL SHRT	216	40912	1	--	2

MAP-300 Buffers

Usage key: 1 = constant
2 = variable

Name	BID	Description	Positional Characteristics	Written by:	Read by:	Sample Incr't	Sample Type	Number of Samples	Halfword Address	Bus #	Initial Values	Usage
TSRB	2	Input Speech to ANL ZB	--	ADINT	ANLZ (PREMP)	1	RL SHRT	216	41128	1	--	2
TUNLS	3	Unity Vector	(Constant vector)	BBRI6I	ANLZ (PREMP) (SPEC)	0	RL LNG	(216)	0	3	Ones (Constant vector)	1
TSP0	4	Preemphasized speech (current frame)		ANLZ (PREMP)	ANLZ (PITCH)	1	RL LNG	(216)	434	3	--	2
TSP1	5	Preemphasized speech (last frame and current frame)	(see TSP0)	ANLZ (PREMP)	ANLZ (PITCH) (INVPF)	1	RL LNG	432	2	3	zeros	2
TSP	6	Preemphasized speech (last 49 of last frame, all 216 of current frame)	(see TSP0)	ANLZ (THIST) (PREMP)	ANLZ (PITCH)	1	RL LNG	(265)	336	3	--	2
THAMP	7	Hanning window coefficients for 265 sample window	--	BBRI6I	ANLZ (PITCH)	1	RL LNG	265	0	2	265 Hanning Window Coefficients	1
TWSP	8	Windowed speech		ANLZ (PITCH)	ANLZ (PITCH)	1	RL LNG	(265)	866	3	--	2

MAP-300 Buffers

Usage key: 1 - Constant
2 - Variable

Name	BID	Description	Positional Characteristics	Written by:	Read by:	Sample Incr't	Sample Type	Number of Samples	Halfword Address	Bus #	Initial Values	Usage
WSPZ	9	Windowed speech with appended zeros	(see TWSP)	BBN161	ANLZ (PITCH)	1	RL LNG	512	866	3	zeros	2/1
TCOST	10	Cosine table for FFT	--	BBN161	ANLZ (PITCH)	1	RL LNG	256	1890	3	cosine table	1
TXSP	11	Frequency domain representation of TWSP (this buffer contains 256 complex points; a 257th complex point will be assumed present directly following it).	Each complex sample: $\left[\begin{array}{c} \text{real} \\ \text{imag.} \end{array} \right]$ TXSPR(i) TXSPI(i)	ANLZ (PITCH)	ANLZ (PITCH)	1	CMPLX LNG	(256)	2402	3	--	2
TXSPR	12	Real part of TXSP, including 257th point	(see TXSP)	ANLZ (PITCH)	ANLZ (PITCH)	2	RL LNG	257	2402	3	--	2
TXSPI	13	Imaginary part of TXSP, including 257th point	(see TXSP)	ANLZ (PITCH)	ANLZ (PITCH)	2	RL LNG	257	2404	3	--	2
TPSP	14	Complex buffer of TPSPR (this buffer contains 256 complex points; a 257th complex point will be assumed present directly following it).	Each complex sample: $\left[\begin{array}{c} \text{real} \\ \text{imag.} \end{array} \right]$ TPSPR(i) TPSPI(i)	ANLZ (PITCH)	ANLZ (PITCH)	1	CMPLX LNG	(256)	3430	3	--	2
TPSPR	15	Power spectrum of TXSP comprising real part of TPSP, including 257th point.	(see TPSP)	ANLZ (PITCH)	ANLZ (PITCH)	2	RL LNG	257	3430	3	--	1

MAP-300 Buffers

Usage key: 1 = constant
2 = variable

Name	BID	Description	Positional Characteristics	Written by:	Read by:	Sample Incr't	Sample Type	Number of Samples	Halfword Address	Bus #	Initial Values	Usage
TPSPI	16	Zeros, comprising imaginary part of TPSP, including 257th point	(see TPSP)	BBN161	ANLZ (PITCH)	2	RL LNG	257	3432	3	zeros	2
TRP	17	Autocorrelation coefficients for pitch determination (IFFT (TPSP))	--	ANLZ (PITCH)	ANLZ (PITCH)	1	RL LNG	512	4458	3	--	2
TC	18	Pitch predictor coeffs	--	ANLZ (PITCH)	ANLZ (INVPF)	1	RL LNG	3	5482	3	--	2
TRPP	19	Autocorrelation coeffs. -TRP(pitch lag -1), -TRP(pitch lag), -TRP(pitch lag +1)	--	ANLZ (PITCH)	ANLZ (PITCH)	1	RL LNG	3	5488	3	--	2
TWORK	20	Work area for MLWL	--	ANLZ (PITCH)	ANLZ (PITCH)	1	RL LNG	6	5494	3	--	2
TIC	21	Coded pitch predictor coefficients	--	ANLZ (PITCH)	ANLZ (GTHR)	1	FXD LNG	3	5506	3	--	2
TCH	22	Quantized pitch predictor coefficients	$\begin{array}{c} \boxed{1} \leftarrow 6 \rightarrow \boxed{+3} \rightarrow \boxed{+6} \rightarrow \\ \leftarrow \boxed{+1} \boxed{+2} \boxed{+3} \boxed{+4} \boxed{+5} \boxed{+6} \rightarrow \\ \boxed{+1} \boxed{+2} \boxed{+3} \boxed{+4} \boxed{+5} \boxed{+6} \rightarrow \\ \boxed{+1} \boxed{+2} \boxed{+3} \boxed{+4} \boxed{+5} \boxed{+6} \rightarrow \end{array}$	ANLZ (PITCH)	ANLZ (INVPF) (APC)	1	RL LNG	3	5524	3	--	2

MAP-300 Buffers

Usage key: 1 = constant
2 = variable

Name	ID	Description	Positional Characteristics	Written by:	Read by:	Sample Incr't	Sample Type	Number of Samples	Halfword Address	Bus #	Initial Values	Usage
TE10	23	First residual (current frame)		ANLZ (INVPF)	ANLZ (SPEC)	1	RL LMG	(216)	5554	3	--	2
TE11	24	First residual (last 6 samples of current frame)	(see TE10)	ANLZ (INVPF)	ANLZ (INVSF)	1	RL LMG	(6)	5974	3	--	2
TE1	25	First residual (last 6 samples of last frame, plus all of current frame)	(see TE10)	ANLZ (INVPF) (THIST)	ANLZ (INVSF)	1	RL LMG	222	5542	3	zeros	2
TI0M1	26	Hanning window coeffs for 216 sample window	--	BHR161	ANLZ (SPEC)	1	RL LMG	216	530	2	216 Hanning Window Coefficients	1
TF1	27	Windowed First Residual	--	ANLZ (SPEC)	ANLZ (SPEC)	1	RL LMG	216	962	2	--	2
TR5	28	Autocorrelation coeffs for Spectral Analysis	--	ANLZ (SPEC)	ANLZ (SPEC)	1	RL LMG	7	5986	3	--	2
TI0M1	29	LAMBDA*MI(n) for HFC	--	BHR161	ANLZ (SPEC)	1	RL LMG	3	6000	3	$\begin{pmatrix} .035 \\ .035 \end{pmatrix} * \begin{pmatrix} .375 \\ -.25 \\ .0625 \end{pmatrix}$	1

MAP-300 Buffers

Usage key: 1 = constant
2 = variable

Name	BID	Description	Positional Characteristics	Written by:	Read by:	Sample Incr't	Sample Type	Number of Samples	Halfword Address	Bus #	Initial Values	Usage
TRSP	30	HFC'd Autocorrelation Coeffs.	--	ANLZ (SPEC)	ANLZ (SPEC)	1	RL LNG	7	6006	3	--	2
TKE	31	Reflection Coeffs. Error terms from PHL (6)	--	ANLZ (SPEC)	ANLZ (SPEC)	1	RI LNG	12	6020	3	--	2
TKH	32	Quantized Reflection Coeffs.	--	ANLZ (SPEC)	ANLZ (SPEC)	1	RL LNG	6	6044	3	--	2
TIK	33	Coded Reflection Coeffs.	--	ANLZ (SPEC)	ANLZ (GTHR)	1	FXD LNG	6	6056	3	--	2
TAH (THH1)	34	Predictor Coeffs (also start of filter coeff block)	(see TCH)	ANLZ (SPEC)	ANLZ (SPEC) (NS) (APC)	1	RL LNG	(6)	5512	3	--	?
TAHR	35	Reverse Buffer of TAH with last element = 1.	(see TCH)	ANLZ (SPEC)	ANLZ (SPEC) (NS) (APC)	-1	RL LNG	7	5510	3	ones	2
TEP	36	6 cond residual	--	ANLZ (THV5f)	ANLZ (GATH)	1	PL LNG	216	6062	3	--	?

MAP-300 Buffers

Usage key: 1 - Constant
2 - Variable

Name	BID	Description	Positional Characteristics	Written by:	Read by:	Sample Increment	Sample Type	Number of Samples	Halfword Address	Bus #	Initial Values	Usage
IFAC	37	Factors for Noise Shaping Analysis	--	BBN161	ANLZ (NS)	1	RL LNG	6	6494	3	.684128 .468032 .320194 .219054 .149861 .102524	1
TANS	38	Noise shaping filter Coeffs	(see TCH)	ANLZ (NS)	ANLZ (APC)	1	RL LNG	6	5530	3	--	2
TGFAC	39	Quantizer Scale factors alternating with their reciprocals	TGFAC: GFAC(1), 1/GFAC(1) GFAC(2), 1/GFAC(2) GFAC(3), 1/GFAC(3)	ANLZ (GAIN)	ANLZ (APC)	1	RL LNG	6	6506	3	--	2
TIMG	40	Coded Delta Gains	--	ANLZ (GAIN)	ANLZ (GTHR)	1	FXD LNG	3	6518	3	--	2
TUD	41	Unquantized, normalized Residual (for debugging only)	--	ANLZ (APC)	SYNZ (SPECF)	1	RL LNG	216	1394	2	--	2
TUJ	42	History for Noise-Shaping Filter	--	BBN161 ANLZ (APC)	ANLZ (APC)	1	RL LNG	6	6522	3	zeros	2
TUH	43	History for Spectral Filter	--	BBN161 ANLZ (APC)	ANLZ (APC)	1	RL LNG	6	6534	3	zeros	2

MAP-300 Buffers

Usage key: 1 = constant
2 = variable

Name	BID	Description	Positional Characteristics	Written by:	Read by:	Sample Incr't	Sample Type	Number of Samples	Halfword Address	Bus #	Initial Values	Usage
TRI0	44	APC Pitch Filter History (Current frame)		ANLZ (APC)	ANLZ (APC)	1	RL LMG	(216)	6978	3	--	2
TRI1	45	APC Pitch Filter History (last frame, current frame)	(see TRI0)	BBR161 ANLZ (APC) (TH151)	ANLZ (APC)	1	RL LMG	432	6546	3	zeros	2
TSRKA (TI0A)	46	Coded transmission data for ANLZA: TI0A, TI6, TI0G, TI1, TIC, TIK	--	ANLZ (APC) (GTHR)	PROPAR, PRORES	1	FXD LMG	230	41344	1	zeros	2
TSRKB (TI0B)	47	Coded transmission data for ANLZB: TI0B, TI6, ...	--	ANLZ (APC) (GTHR)	PROPAR, PRORES	1	FXD LMG	230	41574	1	zeros	2
TBTA	--	TBTS A Buffer	--	PROPAR, PRORES	TMINT	1	FXD LMG	522	41804	1	--	2
TBTB	--	TBTS B Buffer	--	PROPAR, PRORES	TMINT	1	FXD LMG	522	42326	1	--	2
TBTU	--	TBTS C Buffer	--	BBR161	TMINT	1	FXD LMG	522	42848	1	Bitstream equivalent of "Silence" (coded energy 0)	1

MAP-300 Buffers

Usage key: 1 - Constant
2 - variable

Name	BID	Description	Positional Characteristics	Written by:	Read by:	Sample Incr't	Sample Type	Number of Samples	Halfword Address	Bus #	Initial Values	Usage
IMDMA	--	MODEM A Buffer	--	TMINT	MODEM Scroll Pgm	1	FXD LRG	522	43370	1	Bitstream equivalent of "Silence" (coded energy - 0) sync bit - 0	2
IMDMB	--	MODEM B Buffer	--	TMINT	MODEM Scroll Pgm	1	FXD LRG	522	43892	1	Bitstream equivalent of "Silence" (coded energy - 0) sync bit - 1	2
RMDMA	--	MODEM A Buffer	--	Modem Scroll Pgm	RMINT	1	FXD LRG	522	44414	1	--	2
RMDMB	--	MODEM B Buffer	--	Modem Scroll Pgm	RMINT	1	FXD LRG	522	44936	1	--	2
RMDMC	--	MODEM C Buffer	--	Modem Scroll Pgm	RMINT	1	FXD LRG	522	45458	1	--	2
MDMA	--	DATA A Buffer	--	RMINT	DECODE/B, CORPAR	1	FXD LRG	522	45980	1	--	2
MDMB	--	DATA B Buffer	--	RMINT	DECODE/B, CORPAR	1	FXD LRG	522	46502	1	--	2

MAP-300 Buffers

Usage key: 1 - constant
2 - variable

Name	AID	Description	Positional Characteristics	Written by:	Read by:	Sample Incr't	Sample Type	Number of Samples	Halfword Address	Bus #	Initial Values	Usage
RRA (RRAA)	48	Received decoded transmission data: RRAA, RRA, RRAH, RRAH, RRAH, RRAH	--	DECODEA/B, CORPAR	SYNZ (SPECF) (DEAL)	1	RL LING	230	47024	1	zeros	2
RRA (RRAA)	49	Received decoded transmission data: RRAA, RRA, RRAH, RRAH, RRAH, RRAH	--	DECODEA/B, CORPAR	SYNZ (SPECF) (DEAL)	1	RL LING	230	47484	1	zeros	2
RRAH	50	Received Decoded $\sqrt{\text{Delta Gains}}$	--	SYNZ (DEAL)	SYNZ (SPECF)	1	RL LING	3	7410	3	--	2
RRA	51	Received Decoded Pitch Predictor Coefficients	--	SYNZ (DEAL)	SYNZ (PITF)	1	RL LING	3	7416	3	--	2
RRA	52	Received Decoded Reflection Coefficients	--	SYNZ (DEAL)	SYNZ (SPECF)	1	RL LING	8	7422	3	zeros	2
RRA	53	Scaled Residual Samples	--	SYNZ (SPECF)	SYNZ (SPECF)	1	RL LING	216	1826	2	--	2
RRA	54	Spectral Synthesis Filter Memory	--	SYNZ (SPECF)	SYNZ (SPECF)	1	RL LING	8	7438	3	zeros	2

MAP-300 Buffers

Usage key: 1 = Constant
2 = Variable

Name	BID	Description	Positional Characteristics	Written by:	Read by:	Sample Incr't	Sample Type	Number of Samples	Halfword Address	Bus #	Initial Values	Usage
RPH	55	Output of Spectral Synthesis Filter	--	SYN/ (SPCTF)	SYN/ (PTF)	1	RL LRG	216	7454	3	--	2
RPH0	56	Output of Pitch Synthesis Filter (current frame)		SYN/ (PTF)	SYN/ (DEMP) (RHTST)	1	RL LRG	(216)	2690	2	--	2
RPH	57	Output of Pitch Synthesis Filter (last frame and current filter)	(see RPH0)	SYN/ (RHTST) (PTF)	SYN/ (PTF)	1	RL LRG	432	2258	2	zeros	2
RPH1A (RPH1A)	58	Preemphasized Synthetic Speech	--	SYN/ (DEMP)	DAHT	1	RL SHRT	216	47944	1	--	2
RPH2 (RPH2)	59	Decompanded Synthetic Speech	--	SYN/ (DEMP)	DAHT	1	RL SHRT	216	48160	1	--	2
RPH3	--	Synthetic "silence" data	--	RHTST	DAHT	1	RL SHRT	216	48376	1	zeros	1
RPH4A	--	D/A Output to ADM	--	DAHT	ADM Suball Program	1	RL SHRT	216	48592	1	zeros	2

MAP-300 Buffers

Usage key: 1 - constant
2 - variable

Name	BID	Description	Positional Characteristics	Written by:	Read by:	Sample Incr't	Sample Type	Number of Samples	Halfword Address	Bus #	Initial Values	Usage
MDAIB	--	D/A Output to AUM	--	DAINT	AUM Scroll Program	1	RL SHRT	216	48808	1	zeros	2

APPENDIX C
MAP-300 SCALARS

The tables contained in this Appendix describe the characteristics of each user-defined real and integer scalar referenced in the MAP-300 speech coder implementation. The table entry labeled "SID" or "ISID" indicates the real or integer scalar identification number. The entries labeled "Written by:" and "Read by:" indicate subroutines, function lists, or program modules that so access each scalar. (The A/D, D/A, transmitter modem, and receiver modem interrupt service routines are specified in these tables by "ADINT", "DAINT", "TMINT", and "RMINT", respectively.)

MAP-300 REAL SCALARS
Usage key: 1 = constant
2 = variable

Name	SID	Description	Positional Characteristics	Written by:	Read by:	Initial Value	Usage
TALPH	50	Preemphasis filter coefficient	-	BBN161	ANLZ (PREMP)	0.4	1
PPHST	51	Preemphasis filter history	-	ANLZ (PREMP)	ANLZ (PREMP)	0	2
TDCN	52	Negative of current frame's dc term	-	ANLZ (PITCH)	ANLZ (PITCH)	-	2
TPSZ1	53	1 - (Size of Pitch Extraction Buffer)	-	BBN161	ANLZ (PITCH)	1.0 - 265.	1
TMH	54	Pitch lag (in samples)	-	ANLZ (PITCH)	ANLZ (TRVFF) (AFC)	-	2
TIM	55	Coded pitch lag (Integer in left inward)	-	ANLZ (PITCH)	ANLZ (GTER)	-	2
PI4	56	Constant 14	-	BBN161	ANLZ (PITCH)	14.	1

MAP-300 REAL SCALARS

Usage key: 1 = constant
2 = variable

Name	SID	Description	Positional Characteristics	Written by:	Read by:	Initial Value	Usage
TKTHR	57	Threshold value for MEWL and MWLQ6	-	BBN161	ANLZ (PITCH) (SPEC)	.9995	1
TIG	58	Cooled Frame Energy (Integer in left word)	-	ANLZ (GAIN)	ANLZ (GTHR)	-	2
TC2P	59	Pitch Predictor Coefficient for Single Tap Case	-	ANLZ (PITCH)	ANLZ (PITCH)	-	2
T72R	60	Constant: Reciprocal of 72	-	BBN161	ANLZ (APC)	$\frac{1}{72}$	1
T3R	61	Constant: Reciprocal of 3	-	BBN161	ANLZ (APC)	$\frac{1}{3}$	1
Rgll	65	Received Frame Energy	-	SYNZ (DEAL)	SYNZ (SPEC)	-	2
RMI	66	Received Pitch	-	SYNZ (DEAL)	SYNZ (PITCH)	-	2

MAP-300 REAL SCALARS

Usage key: 1 = constant
2 = variable

Name	SID	Description	Positional Characteristics	Written by:	Read by:	Initial Value	Usage
RRCF0	67	Deemphasis filter coefficient SA0	Must be consecutive ↕	BBN161	SYNZ (DEMP)	0.	1
RRCF1	68	Deemphasis filter coefficient SA1	Must be consecutive ↕	BBN161	SYNZ (DEMP)	0.	1
RRCF2	69	Deemphasis filter coefficient SA2	Must be consecutive ↕	BBN161	SYNZ (DEMP)	- ALPHA = -0.4	1
RRCF3	70	Deemphasis filter coefficient SA3	Must be consecutive ↕	BBN161	SYNZ (DEMP)	0.	1
RMY0	71	Deemphasis filter memory Y(-2)	Must be consecutive ↕	SYNZ (DEMP)	SYNZ (DEMP)	0.	2
RMY1	72	Deemphasis filter memory Y(-1)	Must be consecutive ↕	SYNZ (DEMP)	SYNZ (DEMP)	0.	2
RMY2	73	Deemphasis filter memory U(-2)	Must be consecutive ↕	SYNZ (DEMP)	SYNZ (DEMP)	0.	2

MAP-300 REAL SCALARS Usage key: 1 = constant
2 = variable

Name	SID	Description	Positional Characteristics	Written by:	Read by:	Initial Value	Usage
RMY1	74	Decomphasis filter memory 0(-1)	↓ Must be consecutive	SYN2 (DEMP)	SYN2 (DEMP)	0.	2

MAP-300 INTEGER SCALARS

Usage key: 1 = constant
2 = variable

Name	ISID	Description	Positional Characteristics	Written by:	Read by:	Initial Value	Usage
TSRFA	50	TSRA Buffer Status Flag (0-> Empty)	-	ANLZ, ADINT	BBRLP, ADINT	0	2
TSRFB	51	TSRB Buffer Status Flag (0-> Empty)	-	ANLZ, ADINT	BBRLP, ADINT	0	2
TBTF A	52	TBTA Buffer Status Flag (0-> Empty)	-	ANLZ, TMINT	BBNLP, TMINT	0	2
TBTF B	53	TBTB Buffer Status Flag (0-> Empty)	-	ANLZ, TMINT	BBNLP, TMINT	0	2
TTIC	54	MMI output: Number of iterations not completed	Must be consecutive	ANLZ (PITCH)	-	-	2
TEFAG	55	MMI output: Fractional energy E(R)/R(0)	Must be consecutive	ANLZ (PITCH)	-	-	2
-	56	(Unused)	-	-	-	-	-

MAP-300 INTEGER SCALARS

Usage key: 1 = constant
2 = variable

Name	ISID	Description	Positional Characteristics	Written by:	Read by:	Initial Value	Usage
RRTFA	57	RRTFA Buffer Status Flag (0 -> Empty)	-	SYNZ, RMINT	BRNLP, RMINT	0	2
RRTFB	58	RRTFB Buffer Status Flag (0 -> Empty)	-	SYNZ, RMINT	BRNLP, RMINT	0	2
RSRFA	59	RSRFA Buffer Status Flag (0 -> Empty)	-	SYNZ, DAINT	BRNLP, DAINT	0	2
RSRFB	60	RSRFB Buffer Status Flag (0 -> Empty)	-	SYNZ, DAINT	BRNLP, DAINT	0	2
RUR	61	System Run Flag (0 -> Stop)	Must be consecutive ↕	BBRTS, BBRTLE	BBRTS	0	2
-	62	(left unused for MPTM)	Must be consecutive	-	-	-	-
-	63	(Unused)	-	-	-	-	-

MAP-300 INTEGER SCALARS

Usage key: 1 = constant
2 = variable

Name	ISID	Description	Positional Characteristics	Written by:	Read by:	Initial Value	Usage
ADPO	64	TSRA/B Buffer Pointer (0->B; -2->A)	-	ADINT	ADINT	0	2
TSRPO	65	TSRA/B Buffer Pointer (0->B; -2->A)	-	ADINT	ADINT	-2	2
TBPO	66	TBTA/B Buffer Pointer (0->B; -2->A)	-	TMINT	TMINT	0	2
TMPO	67	TMMA/B Buffer Pointer (0->B; -2->A)	-	TMINT	TMINT	0	2
RMPO	68	RMMA/B/C Buffer Pointer (0->C; -2->B; -4->A)	-	RMINT	RMINT	0	2
RBPO	69	RBTA/B Buffer Pointer (0->B; -2->A)	-	RMINT	RMINT	-2	2
RRPO	70	RRMA/B Buffer Pointer (0->B; -2->A)	-	DAINT	DAINT	0	2

MAP-300 INTEGER SCALARS

Usage key: 1 = constant
2 = variable

Name	ISID	Description	Positional Characteristics	Written by:	Read by:	Initial Value	Usage
DAI0	71	DATA/B Buffer Pointer (0->B; -2->A)	-	DAINT	DAINT	0	2
TAI0	72	A/D Frame Discard Counter	-	ADINT	BBN16E	0	2
TNFC	73	TIMEOUT Frame Counter	-	TNINT	BBN16E	0	2
RMFC	74	RMODEM Frame Discard Counter	-	RMINT	BBN16E	0	2
BSRR	75	BSINK Data- Not-Ready Counter	-	DAINT	BBN16E	0	2
TCCTR	76	Transmitter Frame Counter	-	ADINT	BBN16E	0	2
RCCTR	77	Receiver Frame Counter	-	RMINT	BBN16E	0	2

MAP-300 INTEGER SCALARS

Usage key: 1 = constant
2 = variable

Name	ISID	Description	Positional Characteristics	Written by:	Read by:	Initial Value	Usage
RLCTR	78	Receiver Lost-Sync Counter	-	RMINT	BBN16E	0	2
RRHRK	79	Local Handset On-Block State (0->Handset Off->Block)	-	RMINT	ADINT, BBN16E	0	2
RRSYN	80	State of Receiver Sync 0->Lost Sync -1->Searching Sync +1->Found Sync	-	RMINT	RMINT BBN16E	0	2
RROFF	81	Beginning-of-Frame Offset (Sync bit position in RMODEM buffer)	Must be consecutive ↕ Must be consecutive	RMINT	RMINT BBN16E	-	2
-	82	Left unused for R-CTM					
ALCTR	83	A/D (ADAM) Interrupt Counter	-	Device 23, line 1 interrupt service routine	BBN16E	0	2
TMCTR	84	TMODEM (TUG-2) Interrupt Counter	-	Device 16, line 1 interrupt service routine	BBN16E	0	2

MAP-300 INTEGER SCALARS

Usage key: 1 = constant
2 = variable

Name	ISIO	Description	Positional Characteristics	Written by:	Read by:	Initial Value	Usage
RINTPR	85	PRINTER (IOS-2) Interrupt Counter	-	Device 16 line 2 interrupt service routine	BBN16E	0	2
DSACTW	86	D.A. (AOM) Interrupt Counter	-	Device 22 line 1 interrupt service routine	BBN16E	0	2
TADPEK	87	A/D Peak Input Level	-	ADPEAK, BBN16E	BBN16E	0	2
-	88- 122	(Unused)	-	-	-	-	-
RRCOR	123	Error-Correction switch (0 > Error Correction)	Must be consecutive ↑	BBN16E	COMPAR	0	2
-	124	(Left unused for MPTM)	Must be consecutive ↑	-	-	-	-
ERRG	125	Set to non-zero to cause channel error stimulation	Must be consecutive ↑	BBN16E	RMINT	0	2

HAP-300 INTEGER SCALARS

Usage key: 1 = constant
2 = variable

Name	ISID	Description	Positional Characteristics	Written by:	Read by:	Initial Value	Usage
-	126	(left unused for MPTM)	↓ Must be consecutive	-	-	-	-
-	127	(unused)	-	-	-	-	-

APPENDIX D
PROGRAM LISTINGS

This Appendix contains listings of the BBN-written (and modified) MAP-300 and PDP-11 (FORTRAN) programs referred to in Sections 2.4 and 3.

	Page
BBN16M.MLI	139
BBN16P.MLI	299
BBN16T.MLI	305
BBN16U.MLI	318
BBN16C.FOR	374
BBN16D.FOR	385
BBN16E.FOR	387
BBN16F.FOR	395
BBN16H.FOR	404
BBN16I.FOR	410
BBN16R.FOR	415
HIST16.FOR	417

T A B L E O F C O N T E N T S

ARRAY FUNCTION DISPATCH TABLE PATCHES	PAGE 4
NON-ARRAY FUNCTION DISPATCH TABLE PATCHES	PAGE 6
SYMBOL DEFINITIONS FOR BBN16 CODING/DECODING TABLE ADDRESSES	PAGE 7
APU, APS, AND CSPD CODE FOR ADDED FUNCTIONS	PAGE 8
APUJ-VLTSY VECTOR LATTICE SYNTHESIS FILTER	PAGE 9
APS3-V328A APS PROGRAM FOR VLTSY	PAGE 13
APU3-VKTOA(V,U) CONVERT REFL*N (K) TO LPC (A) COEFFICIENTS	PAGE 16
APS3-VKTOA(V,U) APS PROGRAM FOR VKTOA	PAGE 18
APU3-PITCH(V,A,U,B,C,D) COMPUTE AND CODE PITCH	PAGE 20
APS3-PITCH(V,A,U,B,C,D)	PAGE 25
MWLQ6(V,A,U,V,M) WEINER-LEVINSON MATRIX SOL'N & P=6 CODING/QUANT.	PAGE 29
MWLQ6-APS PROGRAM	PAGE 34
MWLQSSM - SPECIAL INTERMEDIATE SUPPORT FOR MWLQ6	PAGE 39
MWL(V,I,U,R,V,M)	PAGE 40
MWL - APS PROGRAM	PAGE 42
SMCHMWL - SPECIAL SUPPORT ROUTINE FOR MWL	PAGE 47
APU3 - STAR(V,A,U,V)	PAGE 51
APS3 - STARS - 3-TAP STABILITY CHECK	PAGE 53
APU3-INVP(V,A,U,V) 3-TAP PITCH INVERSE FILTER	PAGE 56
APS3-INVP(V,A,U,V) APS PROGRAM FOR INVP	PAGE 58
APU3-ADPEAK(U) MONITOR PEAK ABS VAL OF INPUT SPEECH	PAGE 60
APS3-ADPEAK	PAGE 62
APUDMP(V) -- DEBUGGING FUNCTION THAT DUMPS THE APU REGISTERS	PAGE 64
APU3 - HFC(V,U,V)	PAGE 65
APS3 - HFC(V,U,V)	PAGE 67
APU3 - GAIN(V,A,U,B,V,C)	PAGE 70
APS3 - GAIN(V,A,U,B,V,C)	PAGE 72
SPECIAL SUPPORT MODULES FOR FEOL & FEOL	PAGE 79
APU3 - FFUSL FOURIER EVEN-ODD SEPARATE LONG (MAP-300)	PAGE 82
APS3 - FFIAL APS PROGRAM FOR FFUSL	PAGE 83
APU3 - APC(V,A,U,V,M,R,S,T) APC (WITH NOISE SHAPING)	PAGE 86
APS3 - APC(V,A,U,V,M,R,S,T) APC (WITH NOISE SHAPING)	PAGE 89
SM3APC SPECIAL BINDING MODULE FOR APC	PAGE 100
APU3 - GTHR(V,A,U,B,V,W)	PAGE 107
APS3 - GTHR(V,A,U,B,V,W)	PAGE 113
APU3 - THIST(V,U,V,M,R,S) SET UP XMITTER FRAME HISTORIES	PAGE 114
APS3 - THIST(V,U,V,M,R,S) APS PROGRAM FOR THIST	PAGE 117
SUNSTH SPECIAL BINDING MODULE FOR THIST	PAGE 118
APU3 - SCIPES(V,A,U,V)	PAGE 120
APS3 - SCIPES(V,A,U,V)	PAGE 123
APU3-PITSYN(V,A,U,V) 3-TAP PITCH SYNTHESIS FILTER	PAGE 127
APS3-PITSYN(V,A,U,V) APS PROGRAM FOR PITSYN	PAGE 129
APU3 - DEAL(V,A,U,B,V,W)	PAGE 131
APS3 - DEAL(V,A,U,B,V,W)	PAGE 132
APU3-DCORZ DISCRETE CORRELATION APU PROGRAM	PAGE 134
APS3-DCORZ APS PROGRAM FOR DISCRETE CORRELATION WITH AUTO ZERO FILL	PAGE 138
SMDCORZ - SPECIAL BINDING MODULE FOR DCORZ	PAGE 141
MPIFFS MODULE TO PROCESS THE MPIFF(ISA,ISB,FLID) FCB	PAGE 144
MPGSC -- G-FLAG SET/CLEAR	PAGE 145
DEBUGGING "BREAKPOINT" FACILITY	PAGE 146
TOP OF EXECUTIVE DEFINITION	PAGE 147

(000001) {[BBNDI<DCA16>BBN16N.MSU.2, 29-Dec-80 14:30:40, Pd: WOLF

DEFINE 'SAP' (SET ADDRESS FIELD) AND 'LAP' (LOAD ADDRESS FIELD) INSTRUCTIONS TO ASSEMBLER. THESE INSTRUCTIONS ARE NEW CSU INSTRUCTIONS AVAILABLE WITH 'REV 18' CSU MICROCODE. THE SETLOAD THE LOW ORDER 17 BITS OF THE REFERENCED FULL-WORD FROM INTO THE INDICATED REGISTER.

OPADD SAF,(1 .LS. 14) + (29 .LS. 8) + \$ZF
OPADD LAF,(1 .LS. 14) + (29 .LS. 8) + \$ZF

DEFINE SYMBOLS FOR ARRAY FUNCTION ASSEMBLIES

FROM THE SNAP-11 EXECUTIVE --- REL. 3.05 --- 5/22/79

AFDTSORG = \$0EB
APASASS = \$245
APASMDR = \$EFA
APSBMDR0 = \$P63
APSBMDR1 = \$P2D
APBSHL = \$24D
APCSCC = \$248
APSDUNE = \$FBI
APSDONPR = \$FBC
APSG0 = \$C
APSG1 = \$D
APSLUC = \$11
APSPUFF = \$10
APSSAID = \$A
APSSCCCR = \$E
APSSSS = \$9
APSSAM0 = \$4
APSSLA = \$1FFC0
APSLUC = \$1FFC0
APSGEM = \$1FFC0
APSR = \$1FFC0

BCT\$AD	=	\$6M4
BCT\$AT	=	\$686
BCT\$BA	=	\$5H2
BCT\$GB	=	\$0
BCT\$PC	=	\$0
BCT\$WS	=	\$3

CK40B\$ = \$1A06
CLR\$GHC1 = \$792
C05\$ = \$319M

CSPUSWDS = \$21FC	(00054)	*
DWY\$ = \$794	(00056)	*
ERRUN\$ = \$1AFA	(00057)	*
FIAS\$ = \$3110	(00058)	*
FDT\$ = \$7E0	(00059)	*
FEOS\$ = \$3040	(00060)	*
FFTS0SZ = \$79A	(00061)	*
FLG\$CLR = \$0	(00062)	*
FLG\$CU = \$4	(00063)	*
FLG\$CI = \$5	(00064)	*
FLG\$C2 = \$6	(00065)	*
FLG\$G3 = \$7	(00066)	*
FLG\$RI = \$11	(00067)	*
FLG\$SET = \$20	(00068)	*
GATHEN\$ = \$1B3C	(00069)	*
HS\$ = 1	(00070)	*
ISVTS\$ = \$502	(00071)	*
LOADSAP = \$1B7E	(00072)	*
LOADSAP1 = \$1B8F	(00073)	*
MSK\$DHB = \$6	(00074)	*
MSK\$LBYT = \$FF00	(00075)	*
MSK\$SBVT = \$00FF	(00076)	*
M\$ = 0	(00077)	*
OQE = \$1B	(00078)	*
S0111\$ = \$35EA	(00079)	*
SHTL\$R5 = \$7AE	(00080)	*
SINCDS\$ = \$23FA	(00081)	*
SIN\$ = \$3192	(00082)	*
SVTS\$ = \$302	(00083)	*
SVTS\$UNI = \$3CE	(00084)	*
SVS\$PLGS = \$1FFCE	(00085)	*
TEM\$0 = \$704	(00086)	*
TUE\$ = \$21FE	(00087)	*
TUE\$PTR = \$208	(00088)	*
VALBWS\$ = \$1CSA	(00089)	*
VSMAZ\$ = \$2070	(00090)	*
W\$ = \$2	(00091)	*
XFL\$01 = \$192C	(00092)	*

PAGE 3: [RUNDJ] <DCA16>BBN16M.MSD.2, 29-Dec-68 14:38:48, Ed: WOLF

```

000007BA (00107) *
          (00108) *
          (00109) *
          (00110) *
00000288 (00111) BL = T0ESPTR
          (00112) *
00288 001073C0 (00113) *
          (00114) *
          (00115) *
          (00116) *
          (00117) *
          (00118) *
          (00119) *

          ZERO = $7BA
          ADDR T0ESCUR(,1)
          RM = 3
          UPDATE TOP OF EXEC POINTER

```

(00120) "ARRAY FUNCTION DISPATCH TABLE PATCHES		
(00121) *		
(00122) *		
(00123) *		
00000900 (00124)	FL = AFDT\$ORC+3*W\$*(132-120)	JFCB 132 (VLTSV)
00000900 (00125)	ADDR VLTSV\$(R7,1)	
00000900 (00126)	ADDR V320W\$(R7,1)	
00000900 (00127)	ADDR CSPUSNOS(,1,0)	
00000900 (00128)		
00000906 (00129)	FL = AFDT\$ORC+3*W\$*(133-120)	JFCB 133 (VKTOA)
00000906 (00130)	ADDR VKTOA\$(R7,1)	
00000906 (00131)	ADDR VKASI(R7,1)	
00000906 (00132)	ADDR CSPUSNOS(,1,0)	
00000906 (00133)		
0000090C (00134)	FL = AFDT\$ORC+3*W\$*(134-120)	JFCB 134 (PITCH)
0000090C (00135)	ADDR PITCH\$(R7,1)	
0000090C (00136)	ADDR PITCH\$(R7,1)	
0000090C (00137)	ADDR CSPUSNOS(,1,0)	
0000090C (00138)		
00000912 (00139)	FL = AFDT\$ORC+3*W\$*(135-120)	JFCB 135 (HML06)
00000912 (00140)	ADDR HMLF\$APU(R7,1)	
00000912 (00141)	ADDR HMLF\$APS(R7,1)	
00000912 (00142)	ADDR HMLQ\$SSM(,1,1)	
00000912 (00143)		
0000091C (00144)	FL = AFDT\$ORC+3*W\$*(142-120)	JFCB 142 (HML)
0000091C (00145)	ADDR HMLW\$SS(R7,1)	
0000091C (00146)	ADDR HMLW\$(R7,1)	
0000091C (00147)	ADDR SOW\$HML(,1,0)	
0000091C (00148)		
0000096C (00149)	FL = AFDT\$ORC+3*W\$*(150-120)	JFCB 150 (STAR)
0000096C (00150)	ADDR STAB\$(R7,1)	
0000096C (00151)	ADDR STAB\$(R7,1)	
0000096C (00152)	ADDR CSPUSNOS(,1,0)	
0000096C (00153)		
000009D2 (00154)	FL = AFDT\$ORC+3*W\$*(167-120)	JFCB 167 (INWPF)
000009D2 (00155)	ADDR INWPF\$(R7,1)	
000009D2 (00156)	ADDR INWPSI(R7,1)	
000009D2 (00157)	ADDR CSPUSNOS(,1,0)	
000009D2 (00158)		
00000A2C (00159)	FL = AFDT\$ORC+3*W\$*(182-120)	JFCB 182 (ADPEAK)
00000A2C (00160)	ADDR ADPEAK(R7,1)	
00000A2C (00161)	ADDR ADPEAK(R7,1)	
00000A2C (00162)	ADDR CSPUSNOS(,1,0)	
00000A2C (00163)		
00000A32 (00164)	FL = AFDT\$ORC+3*W\$*(183-120)	JFCB 183 (APUDHP)
00000A32 (00165)	ADDR APUDHP\$(R7,1)	
00000A32 (00166)	ADDR DUMPS(R7,1)	
00000A32 (00167)	ADDR CSPUSNOS(,1,0)	
00000A32 (00168)		
00000A5C (00169)	FL = AFDT\$ORC+3*W\$*(190-120)	JFCB 190 (HFC)
00000A5C (00170)	ADDR HFC\$(R7,1)	
00000A5C (00171)	ADDR HFC\$(R7,1)	
00000A5C (00172)	ADDR CSPUSNOS(,1,0)	

000000A00 (00173) *	HL = AFDTSORG+3*W5*(196-128)	JFCB 196 (GAIN)
000000A00 (00174)	ADDR GAINS(R7,1)	
000000A00 (00175)	ADDR GAINS(R7,1)	
000000A00 (00176)	ADDR CSPUSNUS(,1,0)	
000000A00 (00177)	ADDR CSPUSNUS(,1,0)	
000000A00 (00178) *	HL = AFDTSORG+3*W5*(197-128)	JFCB 197 (FEOPFL)
000000A00 (00179)	ADDR FEOPFL(R7,1)	
000000A00 (00180)	ADDR FEOPFL(R7,1)	
000000A00 (00181)	ADDR FEOPFL(R7,1)	
000000A00 (00182)	ADDR FEOPFL(R7,1)	
000000A00 (00183) *	HL = AFDTSORG+3*W5*(198-128)	JFCB 198 (FEOPIL)
000000A00 (00184)	ADDR FEOPIL(R7,1)	
000000A00 (00185)	ADDR FEOPIL(R7,1)	
000000A00 (00186)	ADDR FEOPIL(R7,1)	
000000A00 (00187)	ADDR FEOPIL(R7,1)	
000000A00 (00188) *	HL = AFDTSORG+3*W5*(199-128)	JFCB 199 (APC)
000000A00 (00189)	ADDR APC(R7,1)	
000000A00 (00190)	ADDR APC(R7,1)	
000000A00 (00191)	ADDR APC(R7,1)	
000000A00 (00192)	ADDR APC(R7,1)	
000000A00 (00193) *	HL = AFDTSORG+3*W5*(200-128)	JFCB 200 (GTHR)
000000A00 (00194)	ADDR GTHR(R7,1)	
000000A00 (00195)	ADDR GTHR(R7,1)	
000000A00 (00196)	ADDR GTHR(R7,1)	
000000A00 (00197)	ADDR GTHR(R7,1)	
000000A00 (00198) *	HL = AFDTSORG+3*W5*(201-128)	JFCB 201 (THIST)
000000A00 (00199)	ADDR THIST(R7,1)	
000000A00 (00200)	ADDR THIST(R7,1)	
000000A00 (00201)	ADDR THIST(R7,1)	
000000A00 (00202)	ADDR THIST(R7,1)	
000000A00 (00203) *	HL = AFDTSORG+3*W5*(202-128)	JFCB 202 (SCLRES)
000000A00 (00204)	ADDR SCLRES(R7,1)	
000000A00 (00205)	ADDR SCLRES(R7,1)	
000000A00 (00206)	ADDR SCLRES(R7,1)	
000000A00 (00207)	ADDR SCLRES(R7,1)	
000000A00 (00208) *	HL = AFDTSORG+3*W5*(203-128)	JFCB 203 (PITSYN)
000000A00 (00209)	ADDR PITSYN(R7,1)	
000000A00 (00210)	ADDR PITSYN(R7,1)	
000000A00 (00211)	ADDR PITSYN(R7,1)	
000000A00 (00212)	ADDR PITSYN(R7,1)	
000000A00 (00213) *	HL = AFDTSORG+3*W5*(212-128)	JFCB 212 (DEAL)
000000A00 (00214)	ADDR DEAL(R7,1)	
000000A00 (00215)	ADDR DEAL(R7,1)	
000000A00 (00216)	ADDR DEAL(R7,1)	
000000A00 (00217)	ADDR DEAL(R7,1)	
000000A00 (00218) *	HL = AFDTSORG+3*W5*(213-128)	JFCB 213 (DCURZ)
000000A00 (00219)	ADDR DCURZ(R7,1)	
000000A00 (00220)	ADDR DCURZ(R7,1)	
000000A00 (00221)	ADDR DCURZ(R7,1)	
000000A00 (00222)	ADDR DCURZ(R7,1)	
000000A00 (00223) *	HL = AFDTSORG+3*W5*(214-128)	JFCB 214 (DCURZ)
000000A00 (00224)	ADDR DCURZ(R7,1)	
000000A00 (00225)	ADDR DCURZ(R7,1)	
000000A00 (00226)	ADDR DCURZ(R7,1)	
000000A00 (00227)	ADDR DCURZ(R7,1)	
000000A00 (00228)	ADDR DCURZ(R7,1)	
000000A00 (00229)	ADDR DCURZ(R7,1)	
000000A00 (00230)	ADDR DCURZ(R7,1)	
000000A00 (00231)	ADDR DCURZ(R7,1)	
000000A00 (00232)	ADDR DCURZ(R7,1)	
000000A00 (00233)	ADDR DCURZ(R7,1)	

```

PAGE 6: [BND]DCAL6>DBW16M.NSU.2, 29-Dec-88 14:38:48, Ed: VMF
NON-ARRAY FUNCTION DISPATCH TABLE PATCHES

(00224) *NON-ARRAY FUNCTION DISPATCH TABLE PATCHES
(00225) *
(00226) *
(00227) *
(00228) *
(00229) *
(00230) *
(00231) *
(00232) *
(00233) *

00000001 (00228) JL = FDT$ + (WS * 105) JFCB 105 (MPIP)
00000002 (00229) ADDR MPIP$
00000003 (00230) JL = FDT$ + (WS * 106) JFCB 106 (MPGSC)
00000004 (00231) ADDR MPGSC
00000005 (00232)
00000006 (00233)

```


7: (BMD) (DCAL) > BBN16M-MSU.2, 29-Dec-88 14:30:40, Ed: WULF
SYNRM DEFINITIONS FOR BBN16 CODING/DECODING TABLE ADDRESSES

(00234) *SYNRM DEFINITIONS FOR BBN16 CODING/DECODING TABLE ADDRESSES
(00235))
(00236)) TABLES RESIDE IN BUS1, STARTING AT \$9800.
(00237)) TABLE CONTENTS ARE DEFINED IN MODULE "BBN16T".
(00238))

CODING TABLES

00009000 (00239) CTHC1=\$9800
00009010 (00240) CTHC2=CTHC1+ 8*W\$
00009020 (00241) CTHC3=CTHC1
00009030 (00242) CTHK1=CTHC2+16*W\$
00009040 (00243) CTHK2=CTHC1+64*W\$
00009050 (00244) CTHK3=CTHC2+32*W\$
00009060 (00245) CTHK4=CTHC3+16*W\$
00009070 (00246) CTHK5=CTHC4+16*W\$
00009080 (00247) CTHK6=CTHC5+16*W\$
00009090 (00248) CTHG =CTHC6+16*W\$
00009100 (00249) CTHDG=CTHC +64*W\$
00009110 (00250) CTHU =CTHDG+ 4*W\$
00009120 (00251) DVLC1=CTHU + 4*W\$
00009130 (00252) DVLC2=DVLC1+ 8*W\$
00009140 (00253) DVLC3=DVLC1
00009150 (00254) DVLK1=DVLC2+16*W\$
00009160 (00255) DVLK2=DVLK1+64*W\$
00009170 (00256) DVLK3=DVLC2+32*W\$
00009180 (00257) DVLK4=DVLC3+16*W\$
00009190 (00258) DVLK5=DVLC4+16*W\$
00009200 (00259) DVLK6=DVLC5+16*W\$
00009210 (00260) DVLC =DVLC6+16*W\$
00009220 (00261) DVLDG=DVLC +64*W\$
00009230 (00262) DVLU =DVLDG+ 4*W\$
00009240 (00263) DVLUF=DVLU + 4*W\$

DECODING TABLES

PAGE 8: (00NDJ00CA16>R0016M.N50.2, 29-Dec-80 14:30:40, Ed: WOLF
APU, APS, AND CSPU CODE FOR ADDED FUNCTIONS

(00264) *APU, APS, AND CSPU CODE FOR ADDED FUNCTIONS
(00265) *
(00266) * (FUNCTIONS ARE ORDERED BY PCN NUMBER, WITH
(00267) * ARRAY FUNCTIONS FIRST, THEN NON-ARRAY FUNCTIONS.)
(00268) *
(00269) *
(00270) * SET START ADDRESS FOR ADDED FUNCTIONS.
(00271) * REL 3.5 LEAVES EMPTY SPACE (ON BUS 1) AT:
(00272) * \$3000 - \$4000
(00273) * \$4000 - \$4A00
(00274) * \$5000 - \$C000
(00275) * PUT ADDED FUNCTIONS IN THIS SPACE.
(00276) *
00005D00 (00277) * BL = \$5000
00005D00 (00278) *

PAGE 9: (RRND)COCAL6>RRN16M.NSH.2, 29-DEC-80 14:38:40, FD: WULF
APU3-VLTSY VECTOR LATTICE SYNTHESIS FILTER

```

(00279) * APU3-VLTSY VECTOR LATTICE SYNTHESIS FILTER
(00280) *
(00281) *
(00282) * CUCED BY LCUSELL, 3/79
(00283) *
(00284) * BINDS TO APS3-V3200
(00285) * PERFORMS LATTICE SYNTHESIS FILTER, USING REFL*N COEFFS
(00286) * IMPLEMENTS THE FOLLOWING FORTRAN CODE:
(00287) *
(00288) * DU 20 I=1,M
(00289) *
(00290) * F(7)=W(1)-G(7)*K(8)
(00291) * DO 10 J=6,0,-1
(00292) * F(J)=F(J+1)-G(J)*K(J+1)
(00293) * G(J+1)=G(J)+F(J)*K(J+1)
(00294) * G(8)=F(8)
(00295) * V(1)=F(8)
(00296) *
(00297) *
(00298) * THERE ARE THREE INPUT VECTORS:
(00299) * K (LENGTH 8) REFLECTION COEFFICIENTS (V BID)
(00300) * G (LENGTH 8) FILTER MEMORY (U BID)
(00301) * W (LENGTH M) INPUT RESIDUAL SAMPLES (W BID)
(00302) * THERE ARE TWO OUTPUT ARRAYS:
(00303) * G AS BEFORE
(00304) * V OUTPUT SYNTHETIC SPEECH SAMPLES (LENGTH M) (V BID)
(00305) *
(00306) * INPUT STREAM:
(00307) * G(7)
(00308) * G(6)
(00309) * ...
(00310) * G(8)
(00311) * K(8)
(00312) * K(7)
(00313) * ...
(00314) * K(1)
(00315) * W(1)
(00316) * W(2)
(00317) * ...
(00318) * W(M).
(00319) *
(00320) * OUTPUT STREAM:
(00321) * V(1)
(00322) * V(2)
(00323) * ...
(00324) * V(M)
(00325) * G(7)
(00326) * G(6)
(00327) * ...
(00328) * G(8).
(00329) *
(00330) * REGISTER USAGE:
(00331) *

```

```

(00332) *LEFT      RIGHT
(00333) *          M0=G0  M0=G0
(00334) *          M1=G1  M1=G1
(00335) *          M2=G2  M2=G2
(00336) *          M3=G3  M3=G3
(00337) *          M4=G4  M4=G4
(00338) *          M5=G5  M5=G5
(00339) *          M6=G6  M6=G6
(00340) *          M7=G7  M7=G7
(00341) *          M8=G8  M8=G8
(00342) *          M9=G9  M9=G9
(00343) *          M10=G10 M10=G10
(00344) *          M11=G11 M11=G11
(00345) *          M12=G12 M12=G12
(00346) *          M13=G13 M13=G13
(00347) *          M14=G14 M14=G14
(00348) *          M15=G15 M15=G15
(00349) *          M16=G16 M16=G16
(00350) *          M17=G17 M17=G17
(00351) *          M18=G18 M18=G18
(00352) *          M19=G19 M19=G19
(00353) *          M20=G20 M20=G20
(00354) *          M21=G21 M21=G21
(00355) *          M22=G22 M22=G22
(00356) *          M23=G23 M23=G23
(00357) *          M24=G24 M24=G24
(00358) *          M25=G25 M25=G25
(00359) *          M26=G26 M26=G26
(00360) *          M27=G27 M27=G27
(00361) *          M28=G28 M28=G28
(00362) *          M29=G29 M29=G29
(00363) *          M30=G30 M30=G30
(00364) *          M31=G31 M31=G31
(00365) *          M32=G32 M32=G32
(00366) *          M33=G33 M33=G33
(00367) *          M34=G34 M34=G34
(00368) *          M35=G35 M35=G35
(00369) *          M36=G36 M36=G36
(00370) *          M37=G37 M37=G37
(00371) *          M38=G38 M38=G38
(00372) *          M39=G39 M39=G39
(00373) *          M40=G40 M40=G40
(00374) *          M41=G41 M41=G41
(00375) *          M42=G42 M42=G42
(00376) *          M43=G43 M43=G43
(00377) *          M44=G44 M44=G44
(00378) *          M45=G45 M45=G45
(00379) *          M46=G46 M46=G46
(00380) *          M47=G47 M47=G47
(00381) *          M48=G48 M48=G48
(00382) *          M49=G49 M49=G49
(00383) *          M50=G50 M50=G50
(00384) *          M51=G51 M51=G51
(00385) *          M52=G52 M52=G52
(00386) *          M53=G53 M53=G53
(00387) *          M54=G54 M54=G54
(00388) *          M55=G55 M55=G55
(00389) *          M56=G56 M56=G56
(00390) *          M57=G57 M57=G57
(00391) *          M58=G58 M58=G58
(00392) *          M59=G59 M59=G59
(00393) *          M60=G60 M60=G60
(00394) *          M61=G61 M61=G61
(00395) *          M62=G62 M62=G62
(00396) *          M63=G63 M63=G63
(00397) *          M64=G64 M64=G64
(00398) *          M65=G65 M65=G65
(00399) *          M66=G66 M66=G66
(00400) *          M67=G67 M67=G67
(00401) *          M68=G68 M68=G68
(00402) *          M69=G69 M69=G69
(00403) *          M70=G70 M70=G70
(00404) *          M71=G71 M71=G71
(00405) *          M72=G72 M72=G72
(00406) *          M73=G73 M73=G73
(00407) *          M74=G74 M74=G74
(00408) *          M75=G75 M75=G75
(00409) *          M76=G76 M76=G76
(00410) *          M77=G77 M77=G77
(00411) *          M78=G78 M78=G78
(00412) *          M79=G79 M79=G79
(00413) *          M80=G80 M80=G80
(00414) *          M81=G81 M81=G81
(00415) *          M82=G82 M82=G82
(00416) *          M83=G83 M83=G83
(00417) *          M84=G84 M84=G84
(00418) *          M85=G85 M85=G85
(00419) *          M86=G86 M86=G86
(00420) *          M87=G87 M87=G87
(00421) *          M88=G88 M88=G88
(00422) *          M89=G89 M89=G89
(00423) *          M90=G90 M90=G90
(00424) *          M91=G91 M91=G91
(00425) *          M92=G92 M92=G92
(00426) *          M93=G93 M93=G93
(00427) *          M94=G94 M94=G94
(00428) *          M95=G95 M95=G95
(00429) *          M96=G96 M96=G96
(00430) *          M97=G97 M97=G97
(00431) *          M98=G98 M98=G98
(00432) *          M99=G99 M99=G99
(00433) *          M100=G100 M100=G100
(00434) *          M101=G101 M101=G101
(00435) *          M102=G102 M102=G102
(00436) *          M103=G103 M103=G103
(00437) *          M104=G104 M104=G104
(00438) *          M105=G105 M105=G105
(00439) *          M106=G106 M106=G106
(00440) *          M107=G107 M107=G107
(00441) *          M108=G108 M108=G108
(00442) *          M109=G109 M109=G109
(00443) *          M110=G110 M110=G110
(00444) *          M111=G111 M111=G111
(00445) *          M112=G112 M112=G112
(00446) *          M113=G113 M113=G113
(00447) *          M114=G114 M114=G114
(00448) *          M115=G115 M115=G115
(00449) *          M116=G116 M116=G116
(00450) *          M117=G117 M117=G117
(00451) *          M118=G118 M118=G118
(00452) *          M119=G119 M119=G119
(00453) *          M120=G120 M120=G120
(00454) *          M121=G121 M121=G121
(00455) *          M122=G122 M122=G122
(00456) *          M123=G123 M123=G123
(00457) *          M124=G124 M124=G124
(00458) *          M125=G125 M125=G125
(00459) *          M126=G126 M126=G126
(00460) *          M127=G127 M127=G127
(00461) *          M128=G128 M128=G128
(00462) *          M129=G129 M129=G129
(00463) *          M130=G130 M130=G130
(00464) *          M131=G131 M131=G131
(00465) *          M132=G132 M132=G132
(00466) *          M133=G133 M133=G133
(00467) *          M134=G134 M134=G134
(00468) *          M135=G135 M135=G135
(00469) *          M136=G136 M136=G136
(00470) *          M137=G137 M137=G137
(00471) *          M138=G138 M138=G138
(00472) *          M139=G139 M139=G139
(00473) *          M140=G140 M140=G140
(00474) *          M141=G141 M141=G141
(00475) *          M142=G142 M142=G142
(00476) *          M143=G143 M143=G143
(00477) *          M144=G144 M144=G144
(00478) *          M145=G145 M145=G145
(00479) *          M146=G146 M146=G146
(00480) *          M147=G147 M147=G147
(00481) *          M148=G148 M148=G148
(00482) *          M149=G149 M149=G149
(00483) *          M150=G150 M150=G150
(00484) *          M151=G151 M151=G151
(00485) *          M152=G152 M152=G152
(00486) *          M153=G153 M153=G153
(00487) *          M154=G154 M154=G154
(00488) *          M155=G155 M155=G155
(00489) *          M156=G156 M156=G156
(00490) *          M157=G157 M157=G157
(00491) *          M158=G158 M158=G158
(00492) *          M159=G159 M159=G159
(00493) *          M160=G160 M160=G160
(00494) *          M161=G161 M161=G161
(00495) *          M162=G162 M162=G162
(00496) *          M163=G163 M163=G163
(00497) *          M164=G164 M164=G164
(00498) *          M165=G165 M165=G165
(00499) *          M166=G166 M166=G166
(00500) *          M167=G167 M167=G167
(00501) *          M168=G168 M168=G168
(00502) *          M169=G169 M169=G169
(00503) *          M170=G170 M170=G170
(00504) *          M171=G171 M171=G171
(00505) *          M172=G172 M172=G172
(00506) *          M173=G173 M173=G173
(00507) *          M174=G174 M174=G174
(00508) *          M175=G175 M175=G175
(00509) *          M176=G176 M176=G176
(00510) *          M177=G177 M177=G177
(00511) *          M178=G178 M178=G178
(00512) *          M179=G179 M179=G179
(00513) *          M180=G180 M180=G180
(00514) *          M181=G181 M181=G181
(00515) *          M182=G182 M182=G182
(00516) *          M183=G183 M183=G183
(00517) *          M184=G184 M184=G184
(00518) *          M185=G185 M185=G185
(00519) *          M186=G186 M186=G186
(00520) *          M187=G187 M187=G187
(00521) *          M188=G188 M188=G188
(00522) *          M189=G189 M189=G189
(00523) *          M190=G190 M190=G190
(00524) *          M191=G191 M191=G191
(00525) *          M192=G192 M192=G192
(00526) *          M193=G193 M193=G193
(00527) *          M194=G194 M194=G194
(00528) *          M195=G195 M195=G195
(00529) *          M196=G196 M196=G196
(00530) *          M197=G197 M197=G197
(00531) *          M198=G198 M198=G198
(00532) *          M199=G199 M199=G199
(00533) *          M200=G200 M200=G200
(00534) *          M201=G201 M201=G201
(00535) *          M202=G202 M202=G202
(00536) *          M203=G203 M203=G203
(00537) *          M204=G204 M204=G204
(00538) *          M205=G205 M205=G205
(00539) *          M206=G206 M206=G206
(00540) *          M207=G207 M207=G207
(00541) *          M208=G208 M208=G208
(00542) *          M209=G209 M209=G209
(00543) *          M210=G210 M210=G210
(00544) *          M211=G211 M211=G211
(00545) *          M212=G212 M212=G212
(00546) *          M213=G213 M213=G213
(00547) *          M214=G214 M214=G214
(00548) *          M215=G215 M215=G215
(00549) *          M216=G216 M216=G216
(00550) *          M217=G217 M217=G217
(00551) *          M218=G218 M218=G218
(00552) *          M219=G219 M219=G219
(00553) *          M220=G220 M220=G220
(00554) *          M221=G221 M221=G221
(00555) *          M222=G222 M222=G222
(00556) *          M223=G223 M223=G223
(00557) *          M224=G224 M224=G224
(00558) *          M225=G225 M225=G225
(00559) *          M226=G226 M226=G226
(00560) *          M227=G227 M227=G227
(00561) *          M228=G228 M228=G228
(00562) *          M229=G229 M229=G229
(00563) *          M230=G230 M230=G230
(00564) *          M231=G231 M231=G231
(00565) *          M232=G232 M232=G232
(00566) *          M233=G233 M233=G233
(00567) *          M234=G234 M234=G234
(00568) *          M235=G235 M235=G235
(00569) *          M236=G236 M236=G236
(00570) *          M237=G237 M237=G237
(00571) *          M238=G238 M238=G238
(00572) *          M239=G239 M239=G239
(00573) *          M240=G240 M240=G240
(00574) *          M241=G241 M241=G241
(00575) *          M242=G242 M242=G242
(00576) *          M243=G243 M243=G243
(00577) *          M244=G244 M244=G244
(00578) *          M245=G245 M245=G245
(00579) *          M246=G246 M246=G246
(00580) *          M247=G247 M247=G247
(00581) *          M248=G248 M248=G248
(00582) *          M249=G249 M249=G249
(00583) *          M250=G250 M250=G250
(00584) *          M251=G251 M251=G251
(00585) *          M252=G252 M252=G252
(00586) *          M253=G253 M253=G253
(00587) *          M254=G254 M254=G254
(00588) *          M255=G255 M255=G255
(00589) *          M256=G256 M256=G256
(00590) *          M257=G257 M257=G257
(00591) *          M258=G258 M258=G258
(00592) *          M259=G259 M259=G259
(00593) *          M260=G260 M260=G260
(00594) *          M261=G261 M261=G261
(00595) *          M262=G262 M262=G262
(00596) *          M263=G263 M263=G263
(00597) *          M264=G264 M264=G264
(00598) *          M265=G265 M265=G265
(00599) *          M266=G266 M266=G266
(00600) *          M267=G267 M267=G267
(00601) *          M268=G268 M268=G268
(00602) *          M269=G269 M269=G269
(00603) *          M270=G270 M270=G270
(00604) *          M271=G271 M271=G271
(00605) *          M272=G272 M272=G272
(00606) *          M273=G273 M273=G273
(00607) *          M274=G274 M274=G274
(00608) *          M275=G275 M275=G275
(00609) *          M276=G276 M276=G276
(00610) *          M277=G277 M277=G277
(00611) *          M278=G278 M278=G278
(00612) *          M279=G279 M279=G279
(00613) *          M280=G280 M280=G280
(00614) *          M281=G281 M281=G281
(00615) *          M282=G282 M282=G282
(00616) *          M283=G283 M283=G283
(00617) *          M284=G284 M284=G284
(00618) *          M285=G285 M285=G285
(00619) *          M286=G286 M286=G286
(00620) *          M287=G287 M287=G287
(00621) *          M288=G288 M288=G288
(00622) *          M289=G289 M289=G289
(00623) *          M290=G290 M290=G290
(00624) *          M291=G291 M291=G291
(00625) *          M292=G292 M292=G292
(00626) *          M293=G293 M293=G293
(00627) *          M294=G294 M294=G294
(00628) *          M295=G295 M295=G295
(00629) *          M296=G296 M296=G296
(00630) *          M297=G297 M297=G297
(00631) *          M298=G298 M298=G298
(00632) *          M299=G299 M299=G299
(00633) *          M300=G300 M300=G300
(00634) *          M301=G301 M301=G301
(00635) *          M302=G302 M302=G302
(00636) *          M303=G303 M303=G303
(00637) *          M304=G304 M304=G304
(00638) *          M305=G305 M305=G305
(00639) *          M306=G306 M306=G306
(00640) *          M307=G307 M307=G307
(00641) *          M308=G308 M308=G308
(00642) *          M309=G309 M309=G309
(00643) *          M310=G310 M310=G310
(00644) *          M311=G311 M311=G311
(00645) *          M312=G312 M312=G312
(00646) *          M313=G313 M313=G313
(00647) *          M314=G314 M314=G314
(00648) *          M315=G315 M315=G315
(00649) *          M316=G316 M316=G316
(00650) *          M317=G317 M317=G317
(00651) *          M318=G318 M318=G318
(00652) *          M319=G319 M319=G319
(00653) *          M320=G320 M320=G320
(00654) *          M321=G321 M321=G321
(00655) *          M322=G322 M322=G322
(00656) *          M323=G323 M323=G323
(00657) *          M324=G324 M324=G324
(00658) *          M325=G325 M325=G325
(00659) *          M326=G326 M326=G326
(00660) *          M327=G327 M327=G327
(00661) *          M328=G328 M328=G328
(00662) *          M329=G329 M329=G329
(00663) *          M330=G330 M330=G330
(00664) *          M331=G331 M331=G331
(00665) *          M332=G332 M332=G332
(00666) *          M333=G333 M333=G333
(00667) *          M334=G334 M334=G334
(00668) *          M335=G335 M335=G335
(00669) *          M336=G336 M336=G336
(00670) *          M337=G337 M337=G337
(00671) *          M338=G338 M338=G338
(00672) *          M339=G339 M339=G339
(00673) *          M340=G340 M340=G340
(00674) *          M341=G341 M341=G341
(00675) *          M342=G342 M342=G342
(00676) *          M343=G343 M343=G343
(00677) *          M344=G344 M344=G344
(00678) *          M345=G345 M345=G345
(00679) *          M346=G346 M346=G346
(00680) *          M347=G347 M347=G347
(00681) *          M348=G348 M348=G348
(00682) *          M349=G349 M349=G349
(00683) *          M350=G350 M350=G350
(00684) *          M351=G351 M351=G351
(00685) *          M352=G352 M352=G352
(00686) *          M353=G353 M353=G353
(00687) *          M354=G354 M354=G354
(00688) *          M355=G355 M355=G355
(00689) *          M356=G356 M356=G356
(00690) *          M357=G357 M357=G357
(00691) *          M358=G358 M358=G358
(00692) *          M359=G359 M359=G359
(00693) *          M360=G360 M360=G360
(00694) *          M361=G361 M361=G361
(00695) *          M362=G362 M362=G362
(00696) *          M363=G363 M363=G363
(00697) *          M364=G364 M364=G364
(00698) *          M365=G365 M365=G365
(00699) *          M366=G366 M366=G366
(00700) *          M367=G367 M367=G367
(00701) *          M368=G368 M368=G368
(00702) *          M369=G369 M369=G369
(00703) *          M370=G370 M370=G370
(00704) *          M371=G371 M371=G371
(00705) *          M372=G372 M372=G372
(00706) *          M373=G373 M373=G373
(00707) *          M374=G374 M374=G374
(00708) *          M375=G375 M375=G375
(00709) *          M376=G376 M376=G376
(00710) *          M377=G377 M377=G377
(00711) *          M378=G378 M378=G378
(00712) *          M379=G379 M379=G379
(00713) *          M380=G380 M380=G380
(00714) *          M381=G381 M381=G381
(00715) *          M382=G382 M382=G382
(00716) *          M383=G383 M383=G383
(00717) *          M384=G384 M384=G384
(00718) *          M385=G385 M385=G385
(00719) *          M386=G386 M386=G386
(00720) *          M387=G387 M387=G387
(00721) *          M388=G388 M388=G388
(00722) *          M389=G389 M389=G389
(00723) *          M390=G390 M390=G390
(00724) *          M391=G391 M391=G391
(00725) *          M392=G392 M392=G392
(00726) *          M393=G393 M393=G393
(00727) *          M394=G394 M394=G394
(00728) *          M395=G395 M395=G395
(00729) *          M396=G396 M396=G396
(00730) *          M397=G397 M397=G397
(00731) *          M398=G398 M398=G398
(00732) *          M399=G399 M399=G399
(00733) *          M400=G400 M400=G400
(00734) *          M401=G401 M401=G401
(00735) *          M402=G402 M402=G402
(00736) *          M403=G403 M403=G403
(00737) *          M404=G404 M404=G404
(00738) *          M405=G405 M405=G405
(00739) *          M406=G406 M406=G406
(00740) *          M407=G407 M407=G407
(00741) *          M408=G408 M408=G408
(00742) *          M409=G409 M409=G409
(00743) *          M410=G410 M410=G410
(00744) *          M411=G411 M411=G411
(00745) *          M412=G412 M412=G412
(00746) *          M413=G413 M413=G413
(00747) *          M414=G414 M414=G414
(00748) *          M415=G415 M415=G415
(00749) *          M416=G416 M416=G416
(00750) *          M417=G417 M417=G417
(00751) *          M418=G418 M418=G418
(00752) *          M419=G419 M419=G419
(00753) *          M420=G420 M420=G420
(00754) *          M421=G421 M421=G421
(00755) *          M422=G422 M422=G422
(00756) *          M423=G423 M423=G423
(00757) *          M424=G424 M424=G424
(00758) *          M425=G425 M425=G425
(00759) *          M426=G426 M426=G426
(00760) *          M427=G427 M427=G427
(00761) *          M428=G428 M428=G428
(00762) *          M429=G429 M429=G429
(00763) *          M430=G430 M430=G430
(00764) *          M431=G431 M431=G431
(00765) *          M432=G432 M432=G432
(00766) *          M433=G433 M433=G433
(00767) *          M434=G434 M434=G434
(00768) *          M435=G435 M435=G435
(00769) *          M436=G436 M436=G436
(00770) *          M437=G437 M437=G437
(00771) *          M438=G438 M438=G438
(00772) *          M439=G439 M439=G439
(00773) *          M440=G440 M440=G440
(00774) *          M441=G441 M441=G441
(00775) *          M442=G442 M442=G442
(00776) *          M443=G443 M443=G443
(00777) *          M444=G444 M444=G444
(00778) *          M445=G445 M445=G445
(00779) *          M446=G446 M446=G446
(00780) *          M447=G447 M447=G447
(00781) *          M448=G448 M448=G448
(00782) *          M449=G449 M449=G449
(00783) *          M450=G450 M450=G450
(00784) *          M451=G451 M451=G451
(00785) *          M452=G452 M452=G452
(00786) *          M453=G453 M453=G453
(00787) *          M454=G454 M454=G454
(00788) *          M455=G455 M455=G455
(00789) *          M456=G456 M456=G456
(00790) *          M457=G457 M457=G457
(00791) *          M458=G458 M458=G458
(00792) *          M459=G459 M459=G459
(00793) *          M460=G460 M460=G460
(00794) *          M461=G461 M461=G461
(00795) *          M462=G462 M462=G462
(00796) *          M463=G463 M463=G463
(00797) *          M464=G464 M464=G464
(00798) *          M465=G465 M465=G465
(00799) *          M466=G466 M466=G466
(00800) *          M467=G467 M467=G467
(00801) *          M468=G468 M468=G468
(00802) *          M469=G469 M469=G469
(00803) *          M470=G470 M470=G470
(00804) *          M471=G471 M471=G471
(00805) *          M472=G472 M472=G472
(00806) *          M473=G473 M473=G473
(00807) *          M474=G474 M474=G474
(00808) *          M475=G475 M475=G475
(00809) *          M476=G476 M476=G476
(00810) *          M477=G477 M477=G477
(00811) *          M478=G478 M478=G478
(00812) *          M479=G479 M479=G479
(00813) *          M480=G480 M480=G480
(00814) *          M481=G481 M481=G481
(00815) *          M482=G482 M482=G482
(00816) *          M483=G483 M483=G483
(00817) *          M484=G484 M484=G484
(00818) *          M485=G485 M485=G485
(00819) *          M486=G486 M486=G486
(00820
```

```

113 05078 00J000E (00385)  MUP\MOV(IQ,M6) 3 ,F5
114 05078 05E05E0 (00386)  MUL(M3,M7)\MUL(M3,M7) 3G7*K0,G6*K7
115 05078 08E0000 (00387)  MOV(IQ,M5)\NOP IK4
116 05078 0000000 (00388)  MUP\MOV(IQ,M5) 3 ,F3
117 05078 08E0000 (00389)  MOV(IQ,M4)\NOP IK2
118 05078 0000000 (00390)  MUP\MOV(IQ,M4) 3 ,K1
119 05078 08E0000 (00391)  MOV(IQ,M4)\NOP IF8=INPUT
120 05078 08E0000 (00392)  MOV(P,M6)\NOP 3G7*K0
121 05078 08E0000 (00393)  SUB(A0,A6)\NOP IF8-G7*K0>F7
122 05078 08E0000 (00394)  MUL(M2,M6)\MOV(A6),MUL(M2,M6) 3G5*K6,G4*K5
123 05078 08E0000 (00395)  MOV(R,EX0)\NOP IF7
124 05078 08E0000 (00396)  MUP\MOV(EXI,M0) 3 ,F7
125 05078 08E0000 (00397)  MUP\SUB(A0,A6) 3 ,F7-G6*K7>F6
126 05078 08E0000 (00398)  MUP\MOV(R,M3) 3 ,F6
127 05078 08E0000 (00399)  MOV(A4),MUL(M1,M5)\MOV(A4),MUL(M3,M7) 3(G5*K6)G3*K4,(G4K5)F6K7
128 05078 08E0000 (00400)  MUP\MOV(R,EX0) 3 ,F6
129 05078 08E0000 (00401)  MOV(EXI,M6)\NOP IF6
130 05078 08E0000 (00402)  SUB(A6,A4)\NOP IF6-G5K6>F5
131 05078 08E0000 (00403)  MUP\MOV(R,EX0) 3 ,F5
132 05078 08E0000 (00404)  MOV(R,M2)\MOV(EXI,M2) 3F5,F5
133 05078 08E0000 (00405)  MOV(A2),MUL(M2,M6)\MOV(A6),MUL(M1,M5) 3(G3K4)F5K6,(F6K7)G2K3
134 05078 08E0000 (00406)  MUP\SUB(A2,A4) 3 ,F5-G4K5>F4
135 05078 08E0000 (00407)  MUP\MOV(R,EX0) 3 ,F4
136 05078 08E0000 (00408)  MOV(EXI,M4)\NOP IF4
137 05078 08E0000 (00409)  SUB(A4,A2)\MOV(M2),ADD(A7,A6) 3F4-G3K4>F3,(F4)G6+P6K7>G7
138 05078 08E0000 (00410)  MOV(A6),MUL(M2,M4)\MOV(A2),MUL(M2,M6) 3(F5K6)G1K2,(G2K3)F4K5
139 05078 08E0000 (00411)  MOV(R,EX0)\MOV(R,EX0) 3F3,G7
140 05078 08E0000 (00412)  MOV(EXI,M3)\MOV(EXI,M4) 3G7(NEW),F3
141 05078 08E0000 (00413)  MOV(M1),ADD(A5,A6)\SUB(A4,A2) 3(F3)G5+FSK6>G6,F3-G2K3>F2
142 05078 08E0000 (00414)  MOV(EXI,M7)\NOP 3G7(NEW)
143 05078 08E0000 (00415)  MOV(A0),MUL(M1,M5)\MOV(A4),MUL(M0,M4) 3(G1K2)F3K4,(F4K5)G0K1
144 05078 08E0000 (00416)  MUP\MOV(R,EX0) 3 ,F2
145 05078 08E0000 (00417)  MOV(EXI,M2)\NOP IF2
146 05078 08E0000 (00418)  MOV(EX0),SUB(A2,M0)\MOV(M1),ADD(A5,A4) 3(G6)F2-G1K2>F1,(F2)G4+
147 05078 08E0000 (00419)  3F4K5>G5
148 05078 08E0000 (00420)  MUP\MOV(EXI,M7) 3 ,G6(NEW)
149 05078 08E0000 (00421)  MUP\MOV(EXI,M3) 3 ,G6(NEW)
150 05078 08E0000 (00422)  MOV(R,EX0)\MOV(R,EX0) 3F1,G5
151 05078 08E0000 (00423)  MOV(R,M0)\MOV(EXI,M2) 3F1,F1
152 05078 08E0000 (00424)  MOV(A4),MUL(M0,M4)\MOV(A0),MUL(M1,M5) 3(F3K4)F1K2,(G0K1)F2K3
153 05078 08E0000 (00425)  ADD(A3,A4)\SUB(A2,M0) 3G3+P3K4>G4,F1-G0K1>F0
154 05078 08E0000 (00426)  MOV(EXI,M5)\NOP 3G5(NEW)
155 05078 08E0000 (00427)  MOV(EXI,M2)\NOP 3G5(NEW)
156 05078 08E0000 (00428)  MOV(R,EX0)\MOV(M,M0) 3G4,F0=00(NEW)
157 05078 08E0000 (00429)  MOV(P,M2)\MOV(A2),MUL(M0,M4) 3F1K2,F0K1
158 05078 08E0000 (00430)  ADD(A1,A2)\MOV(EX0),ADD(A3,A2) 3G1+P1K2>G2,(F0)G2+P2K3>G3
159 05078 08E0000 (00431)  MOV(EXI,M0)\MOV(EXI,M2) 3F0,DUT,74(NEW)
160 05078 08E0000 (00432)  MOV(EXI,M1)\MOV(EXI,M5) 3F0=00,G4(NEW)
161 05078 08E0000 (00433)  MOV(R,EX0)\NOP 3G2
162 05078 08E0000 (00434)  JUMPS(LD00E,F1) 3END IF INPUT USED UP
163 05078 08E0000 (00435)  MOV(IQ,M)\NOP IF8(NEW)=INPUT
164 05078 08E0000 (00436)  MUL(M3,M7)\MOV(A0),MUL(M3,M7) 3G7K0,(F0K1)G6K7
165 05078 08E0000 (00437)  R(A1)\MOV(EX0),ADD(A1,M0) 3F0=00,(G3)G0+P0K1>G1

```

PAGE 12: [RND]<DCA16>RBM16M.MSU.2, 29-Dec-80 14:38:40, Ed: WULF
APU3-VLTSY VECTOR LATTICE SYNTHESIS FILTER

```

A47 05D98 08490849 (00438)
A48 05D92 08510853 (00439)
A49 05D94 08900890 (00440)
A4A 05D96 08A00800 (00441)
A4B 05D98 08510851 (00442)
A4C 05D9A 08500856 (00443)
A4D 05D9C 4E000000 (00444)
A4E 05D9E 10000010 (00445)
      (00446) *
A4F 05DA0 00000000 (00447) *
A50 05DA2 02200038 (00448) LDONE:
A51 05DA4 08510853 (00449)
A52 05DA6 02F002F8 (00450)
A53 05DA8 08900800 (00451)
A54 05DAA 02A0020C (00452)
A55 05DAC 08900800 (00453)
A56 05DAE 0260027C (00454)
A57 05D00 08900800 (00455)
A58 05D02 0800089C (00456)
A59 05D04 08500800 (00457)
A5A 05D06 0800085C (00458)
A5B 05D08 20300800 (00459)
A5C 05D0A 08000800 (00460)
A5D 05D0C 10000000 (00461)
      (00462) *
      (00463) *
      0000005E (00464)
      05D0E (00465)
      (00466) *
      (00467) *

MOV(EXT,M1)\MOV(EXT,M1) JG3(NEW),G2(NEW)
MOV(EXT,A3)\MOV(EXT,A3) JG3(NEW),G2(NEW)
MOV(IN,EXD)\MOV(R,EXD) JG6=FB,G1
MOV(EXT,M0)\NOP JG1(NEW)
MOV(EXT,A1)\MOV(EXT,A1) JG1(NEW),G0(NEW)
MOV(M6)\MUL(M2,M6)\MOV(A6)\MUL(M2,M6) J(G7K0)G5K6,(G6K7)G4K5
SUB(A0,A6)\NOP JF8-G7K8>F7
JUMP(LOOP)

FINISH LAST PART OF LAST LOOP, AND OUTPUT G'S
NOPANDV(P,A0) J ,F0K1
R(A1)\MOV(EXT)\ADD(A1,A0) JF8=GB,(G3)G0+P0K1
MOV(EXT,A3)\MOV(EXT,A3) JG3(NEW),G2(NEW)
MOV(EXT)\R(A7)\MOV(EXT),R(A7) J(G0)G7,(G1)G6
MOV(R,NQ)\NOP JG7 OUT
R(A5)\MOV(OUT),R(A5) JG5,(G6 OUT),G4
MOV(R,NQ)\NOP JG5 OUT
R(A3)\MOV(OUT),R(A3) JG3,(G4 OUT),G2
MOV(R,NQ)\NOP JG3 OUT
NOPANDV(R,NQ) J ,G2 OUT
MOV(EXT,NQ)\NOP JG1 OUT
NOPANDV(EXT,NQ) J ,G0 OUT
CLEAR(RA)\NOP JHALT
NOP
JUMP(0)

VLTSY$SZ=RA-VLTSY$SA
END VLTSY$SZ
EVEN

```

PAGE 13: (RHNJ)COCAL6>BBN16H.MSU.2, 29-Dec-80 14:30:40, Ed: WULF
APS3-V3200 APS PROGRAM FOR VLTSY

```

(00468) * APS3-V3200 APS PROGRAM FOR VLTSY
(00469) *
(00470) *
(00471) *
(00472) *
(00473) *
(00474) *
(00475) *
(00476) *
(00477) *
(00478) *
(00479) *
(00480) *
(00481) *
(00482) *
(00483) *
(00484) *
(00485) *
(00486) *
(00487) *
(00488) *
(00489) *
(00490) *
(00491) *
(00492) *
(00493) *
(00494) *
(00495) *
(00496) *
(00497) *
(00498) *
(00499) *
(00500) *
(00501) *
(00502) *
(00503) *
(00504) *
(00505) *
(00506) *
(00507) *
(00508) *
(00509) *
(00510) *
(00511) *
(00512) *
(00513) *
(00514) *
(00515) *
(00516) *
(00517) *
(00518) *
(00519) *
(00520) *

050BE 00005E2A (00481)
050C0 00000000 (00482)
050C2 0000 (00483)
050C3 0000 (00484)
050C4 00005E22 (00485)
050C5 00005E22 (00486)
050C6 00201D40 (00495)
050C8 00230030 (00496)
050C9 00230030 (00497)
050CA 04401000 (00506)
050CB 06500000 (00507)
050CC 00600000 (00508)
050CD 0A200031 (00509)
050CE 0C00000F (00511)
050CF 0F200601 (00512)
050D0 100A900C (00514)
050D1 12000007 (00516)
050D2 14020004 (00517)
050D3 16100001 (00518)
050D4 16100001 (00519)
050D5 16100001 (00520)

MAP-300 APS PROGRAM FOR VLTSY(Y,U,V,W)
CODED BY KFIELD 3/79

INPUT STREAM: U(7),...,U(0),V(7),...,V(0),W(0),...,W(WBS-1),CP13
OUTPUT STREAM: Y(0),...,Y(WBS-1),U(7),...,U(0),LE03

HEADER BLOCK

EVEN V3200$1 JPTR TO CONSTR. INSTR. BLK.
ADDR 0 JPTR TO SCALAR BLOCK (NO SCALARS)
DATA 0 JNUMBER OF SCALARS
DATA V3200$Z JMODULE SIZE
ADDR V3200$A JPTR TO CHAIN ANCHOR
EVEN

REGIM APS(V3200)

INPUT PROGRAM

JSM(V3200$6,P2) JSET OUTPUT PC
SET(RO) JTURN ON OUTPUT GENERATION

INPUT PCM REGISTER USAGES:
RR0: VECTOR ELEMENT ADDRESSES
RR1: BUFFER SIZES
RR2: BUFFER SPACINGS

GENERATE "U" ADDRESSES

LOAD(BR0,13) JBR0 <= VECTOR U BASE ADDR
LOAD(BR1,MSS) JDUHNY LOAD
LOAD(BR2,MSS) JBR2 <= VECTOR U SPACING
SUBL(BR2,1) JBP2 <= SPACING-1

ADDL(BR0,7) JGEN ADDR OF U(7)
SUBL(BR2,1),JUNPP(B1) JADD (7 * SPACING)
ADD(BR0,193) JBR0 <= ADDR(U7)+SPACING

LOAD(BR1,7) JBR1 <= 7
SUBL(BR0,191,TF) JGEN ADDR (U(1))
SUBL(BR1,1),JUNPP(B2) JLOOP 8 TIMES

```

```

(00521) *
(00522) *
ABC 050DE 18402004 (00523)
APD 050F0 1A500000 (00524)
APE 050F2 1C600000 (00525)
APF 050F4 1E200031 (00526)
A10 050F6 20400017 (00527)
A11 050F8 22201081 (00528)
A12 050FA 2400A00C (00529)
A13 050FC 26500007 (00530)
A14 050FE 2820A004 (00531)
A15 050F0 2A191401 (00532)
(00533) *
(00534) *
(00535) *
(00536) *
(00537) *
(00538) *
(00539) *
A16 050F2 2C403004 (00540)
A17 050F4 2E500000 (00541)
A18 050F6 30200000 (00542)
A19 050F8 3280A006 (00543)
A1A 050FA 341910D1 (00544)
(00545) *
(00546) *
(00547) *
(00548) *
A10 050FC 36200031 (00549)
A1C 050FE 38000020 (00550)
(00551) *
(00552) *
(00553) *
(00554) *
(00555) *
(00556) *
(00557) *
(00558) *
(00559) *
(00560) *
(00561) *
A10 050F0 3A300032 (00562)
(00563) *
(00564) *
(00565) *
A1E 050F2 3C40000A (00566)
A1F 050F4 3E500000 (00567)
A20 050F6 40070000 (00568)
(00569) *
(00570) *
(00571) *
A21 050F8 42703006 (00572)
A22 050FA 44500000 (00573)

GENERATE 'V' ADDRESSES
LOAD(BR0,C33)
LOAD(BR1,M$S)
LOAD(BR2,M$S)
SUBL(BR2,1)
ADDL(BR0,7)
SUBL(BR2,1),JUMPP(B3)
ADD(BR0,C103)
LOAD(BR1,7)
SUB(RN0,C103,TF)
SUBL(BR1,1),JUMPP(B4)

GENERATE 'W' ADDRESSES
LOAD(BR0,C33)
LOAD(BR1,M$S)
SUB(RN0,M$S)
ADD(BR0,C113,TF)
SUBL(BR1,1),JUMPP(B5)

WHEN DONE GENERATING INPUT ADDRESSES, CLEAR RI
CLEAR(RI)
NOP(0)

OUTPUT PROGRAM

OUTPUT PGM REG USAGES:
  RW0: VECTOR ELEMENT ADDRESSES
  RW1: BUFFER SIZES
  RW2: BUFFER SPACINGS
  RW3: SCRATCH REGISTER

TURN ON APU

GEN 'V' ADDRESSES
LOAD(BW0,C03)
LOAD(BW1,M$S)
SUB(BW0,M$S)
GET WBS-1 (GENERATE WBS V'S)
LOAD(BW3,C33)
LOAD(BW1,M$S)

JBR0 <= VECTOR V BASE ADDR
JBR1 <= WBS-1
JBR2 <= W BASE MINUS SPACING
JGEN ADDR(W(1))
JLOOP WBS TIMES

JBR0 <= VECTOR W BASE ADDR
JBR1 <= WBS-1
JBR2 <= W BASE MINUS SPACING
JGEN ADDR(W(1))
JLOOP WBS TIMES

JBR0 <= Y BASE ADDR
JBR1 <= Y BASE MINUS SPACING
JGEN ADDR(W(1))
JLOOP WBS TIMES

```


PAGE 15: (00MD)DCAL16>RBN16M.MS0.2, 29-Dec-80 14:30:40, E3: WOLF
AP53-V3200 APS PROGRAM FOR VLTSY

A23 05E0C 46700000 (00574) *	LOAD(BW3,MSS)	JUNMV LOAD
A24 05E0E 400A0006 (00575) *		
A25 05E10 40112401 (00576) 07	ADD(BW0,C03,TF)	JGEN ADDR(V(1))
	SUBL(BW1,1),JUMPP(07)	FLOOP WBS TIMES
	(00577) *	
	(00578) *	
	(00579) *	
	(00580) *	
A26 05E12 4C401004 (00581)	LOAD(BW0,C13)	JBW0 <= VECTOR U BASE ADDRESS
A27 05E14 4F500000 (00582)	LOAD(BW1,MSS)	JUNMV LOAD
A28 05E16 50600000 (00583)	LOAD(BW2,MSS)	JBW2 <= VECTOR U SPACING
A29 05E18 52210031 (00584) *	SUBL(BW2,1)	JBW2 <= SPACING-1
	(00585) *	
A2A 05E1A 5401003F (00586) 08	ADD(BW0,7)	JGEN ADDR OF U(7)
A2B 05E1C 56212401 (00587) *	SUBL(BW2,1),JUMPP(08)	JADD (7*SPACING)
	(00588) *	
A2C 05E1E 500A'00C (00589) *	ADD(BW0,C93)	JBW0 <= ADDR(U(7)) + SPACING
	(00590) *	
A2D 05E20 5A500007 (00591) *	LOAD(BW1,7)	JBW1 <= 7
A2E 05E22 5C020004 (00592) 09	SUB(BW0,C91,TF)	JGEN ADDR(U(1))
A2F 05E24 5E112EH1 (00593) *	SUBL(BW1,1),JUMPP(09)	FLOOP 8 TIMES
	(00594) *	
	(00595) *	
	(00596) *	
A30 05E26 60200030 (00597) *	CLEAR(00)	JHALT OUTPUT
A31 05E28 62000020 (00598) *	NOP(0)	
	(00599) *	
05E2A 00005E22 (00600) V3200SA=0C	END 0A-1	JASSIGN VALUE TO CHAIN ANCHOR
	(00601) *	JEND OF MODULE
	(00602) *	
	(00603) *	
	(00604) *	
05E2A 00000000 (00605) V3200ST DATA 14F'0.0'		
...		
00000000 (00606) V3200SZ=HL-V3200S		JMODULE SIZE
	(00607) *	

```

PAGE 17: (00ND)10CA16>00016M.N50.2, 29-Dec-88 14:38:48, Ed: WOLF
        AP03-VRT0A(Y,W) CONVERT REFL-M (K) TO LPC (A) COEFFICIENTS

A05 05E52 00C00020 (006611)      MOV(TQA,M0)      INPUT K(M)
      (00662)  JINNER LOOP
A06 05E54 00F0002C (006633)      MOV(TQA,M0) \ MOV(TQA,M4)      JACM-13(M-L)
A07 05E56 00E00070 (006644)      MOV(TQA,M4) \ MOV(TQA,M0)      JACM-13(L)
A08 05E58 04000400 (006645)      MUL(M0,M4)
A09 05E5A 0010001 (006666)      MOV(P,A1)
A0A 05E5C 41004100 (006677)      ADD(M0,A1)
      (00668)  J
A0B 05E5E 009C009C (006699)      MOV(R,OQ)
A0C 05E60 90160006 (00670)      JUMPC(R2,PV1)
A0D 05E62 90100000 (00671)      JUMPC(R3,OQE)
A0E 05E64 00000000 (00672)      NOP
A0F 05E66 20172037 (00673)      CLEAR(VI)
A10 05E68 901C0004 (00674)      JUMPC(R1,E0)
A11 05E6A 20122032 (00675)      CLEAR(RA)
A12 05E6C 10000000 (00676)      JUMP(0)
      (00677)  VKA0$02=BA-VKA0$5A
      (00678)  END
05E6E

```

JACM-13(M-L) + K(M) = JACM-13(L) \
 JACM-13(L) + K(M) = JACM-13(M-L)
 JACM(M-L) \ ACN(L)
 JUMP IF NOT DONE THIS ITERATION
 WAIT FOR THIS ITERATION'S RESULT TO BE
 BE WRITTEN BEFORE LETTING APS PROCEED
 TO READ OPERANDS FOR NEXT ITERATION
 JUMP IF MORE ITERATIONS YET TO DO
 HALT

PAGE 16: [08N03]DCAL6>08N16M.NSO.2, 29-Dec-88 14:38:46, Ed: WOLF
AP03-VKTOA(V,U) CONVERT REFL-M (K) TO LPC (A) COEFFICIENTS

```

(00600) * AP03-VKTOA(V,U) CONVERT REFL-M (K) TO LPC (A) COEFFICIENTS
(00609) )
(00610) ) J. J. WOLF 6/30/88
(00611) ) BINDS TO APS3-VKTOA
(00612) )
(00613) ) THIS VKTUA IS DIFFERENT FROM THAT IN 08N300.NSO (9.6 KB/S CODE)
(00614) ) IN THAT (1) BOTH INPUT AND OUTPUT BUFFERS ARE FLT PT AND (2) THIS
(00615) ) IS WRITTEN TO HANDLE A GENERAL COEFFICIENT VECTOR (ANY LENGTH).
(00616) )
(00617) ) BUFFER V: OUTPUT, LPC COEFFS: A(1), A(2), ..., A(M). A(0), WHICH IS
(00618) ) 1.0, IS NOT OUTPUT.
(00619) ) BUFFER U: INPUT, REFLECTION COEFFS, K(1), K(2), ..., K(M)
(00620) ) NOTE: BUFFERS V AND U MUST BE COMPACT 32-BIT FLT PT.
(00621) )
(00622) ) THE LPC COEFFICIENTS ARE COMPUTED FROM THE RECURSION:
(00623) ) (1)  $ACM(N) = K(N)$ 
(00624) ) (2)  $ACM(L) = ACM-1(L) + K(M) \cdot ACM-1(M-L)$ 
(00625) )  $M=1,2,...,N$ 
(00626) )  $L=1,2,...,M-1$ 
(00627) ) WHERE C.J INDICATES ITERATION AND (.) INDICATES INDEX.
(00628) ) IN ORDER THAT THE COMPUTATION BE DONE IN PLACE, WE DO EQ. (2) IN
(00629) ) "SYMMETRIC" PAIRS:
(00630) ) (2A)  $ACM(M-L) = ACM-1(M-L) + K(M) \cdot ACM-1(L)$ 
(00631) ) (2B)  $ACM(L) = ACM-1(L) + K(M) \cdot ACM-1(M-L)$ 
(00632) )  $L=1,2,...,(M-1)/2$ 
(00633) )
(00634) ) APU REGISTER USAGE:
(00635) ) A(0): ACM-1(M-L) ACM-1(L)
(00636) ) A(1): P-A K-A
(00637) ) M(0): K(M)
(00638) ) M(4): ACM-1(L) ACM-1(M-L)
(00639) )
(00640) ) INPUT, OUTPUT STREAMS:
(00641) ) N=1 M=2 L=1
(00642) ) IQ= K(1)CM1, K(2),A(1),A(1)CM1, K(3),A(2),A(1)CM1,
(00643) ) QO= A(1), A(2),A(1),A(1), A(3),A(2),A(1),
(00644) ) M=4 L=1 L=2
(00645) ) K(4),A(3),A(1),A(2),A(2)CM1,...
(00646) ) A(4),A(3),A(1),A(2),A(2), ...C(0)
(00647) ) THE NUMBER OF ITERATIONS IS UBS.
(00648) )
(00649) ) EVEN
(00650) ) DATA VKAUSSA )START ADDRESS
(00651) ) DATA VKAUSSZ )SIZE
(00652) )
(00653) ) VKAU: BEGIN APU(VKAU)
(00654) ) VKAUSSA:
(00655) )
(00656) ) MUV(IQA,OQ) \ NOP
(00657) ) JUMP(CR10,UQE)
(00658) ) NOP
(00659) ) CLEAR(MI)
(00660) ) ITERATION LOOP
(00661) ) MUV(IQA,OQ) \ NOP
(00662) ) ACM(M)=K(M)
(00663) )
(00664) ) JAL1(1)=K(1). END OF M=1 ITERATION.
(00665) ) WAIT FOR M=1 RESULT TO BE WRITTEN
(00666) ) THEN LET APS PROCEED
(00667) )
(00668) ) JAL1(M)=K(M)
(00669) )
(00670) )
(00671) )
(00672) )
(00673) )
(00674) )
(00675) )
(00676) )
(00677) )
(00678) )
(00679) )
(00680) )
(00681) )
(00682) )
(00683) )
(00684) )
(00685) )
(00686) )
(00687) )
(00688) )
(00689) )
(00690) )
(00691) )
(00692) )
(00693) )
(00694) )
(00695) )
(00696) )
(00697) )
(00698) )
(00699) )
(00700) )
(00701) )
(00702) )
(00703) )
(00704) )
(00705) )
(00706) )
(00707) )
(00708) )
(00709) )
(00710) )
(00711) )
(00712) )
(00713) )
(00714) )
(00715) )
(00716) )
(00717) )
(00718) )
(00719) )
(00720) )
(00721) )
(00722) )
(00723) )
(00724) )
(00725) )
(00726) )
(00727) )
(00728) )
(00729) )
(00730) )
(00731) )
(00732) )
(00733) )
(00734) )
(00735) )
(00736) )
(00737) )
(00738) )
(00739) )
(00740) )
(00741) )
(00742) )
(00743) )
(00744) )
(00745) )
(00746) )
(00747) )
(00748) )
(00749) )
(00750) )
(00751) )
(00752) )
(00753) )
(00754) )
(00755) )
(00756) )
(00757) )
(00758) )
(00759) )
(00760) )
(00761) )
(00762) )
(00763) )
(00764) )
(00765) )
(00766) )
(00767) )
(00768) )
(00769) )
(00770) )
(00771) )
(00772) )
(00773) )
(00774) )
(00775) )
(00776) )
(00777) )
(00778) )
(00779) )
(00780) )
(00781) )
(00782) )
(00783) )
(00784) )
(00785) )
(00786) )
(00787) )
(00788) )
(00789) )
(00790) )
(00791) )
(00792) )
(00793) )
(00794) )
(00795) )
(00796) )
(00797) )
(00798) )
(00799) )
(00800) )

```

```

(00679) - APS3-VKTOA(Y,U)      APS PROGRAM FOR VKTDA
(00680) J J. WOLF, 7/2/80
(00681) J
(00682) JHEADER BLCK
(00683) J
(00684) EVEN
(00685) ADDR VKASSI
(00686) ADDR 0
(00687) DATA 0
(00688) DATA VKASSZ
(00689) ADDR VKASSA
(00690) EVEN
(00691) J
(00692) JINPUT PROGRAM
(00693) J BR0: COUNTS ITERATIONS FROM 2 TO UBS
(00694) J BR1: K(M), A(K)
(00695) J BR2: MI=2*(M-1)
(00696) J BR3: M' COUNTS FROM 2*(M-1) BY 2 TO 0 OR -2
(00697) J
(00698) VKASSI:
(00699) LOAD(BR1,C17,TF)
(00700) LOAD(BR0,M$)
(00701) LOAD(BR3,M$)
(00702) SUBL(BR0,1)
(00703) MOV8(BR0,BR0)
(00704) JSN(VKASSO,02)
(00705) SET(R0)
(00706) LOAD(BR2,0)
(00707) SET(MI)
(00708) NOP(0)
(00709) JITERATION LOOP. WE BEGIN WITH THE M=2 ITERATION.
(00710) M4:
(00711) ADDL(BR2,2)
(00712) LOAD(BR1,C17)
(00713) LOAD(BR3,M$)
(00714) ADDH(BR1,BR2,TF)
(00715) LOAD(BR1,C17)
(00716) LOAD(BR3,M$)
(00717) SUB(BR1,M$)
(00718) MOV8(BR3,BR2),JUMP(VK6)
(00719) JIMMER LOOP: L-1,2,...,(M-1)/2
(00720) M5:
(00721) VK6:
(00722) SUBL(BR3,M$)
(00723) JTHE NEXT 3 INSTRUCTIONS ARE UNINTERRUPTIBLE, ENSURING THAT ON THE FINAL
(00724) J LOOP OF THE ITERATION, MI GETS SET 2 APU-CYCLES AFTER THE LAST READ
(00725) J ADDRESS IS GENERATED, SO IT'S SURE TO BE SET BEFORE THE APU CAN TEST
(00726) J IT.
(00727) SUB8(BR1,BR3,TF),JUMP(BA+1)
(00728) SUBL(BR3,2*MI),JUMPR(M5)
(00729) SET(MI)
(00730) NOP(0)
(00731) SUBL(BR0,1),JUMPP(M4)
(00732) J
(00733) J
(00734) J
(00735) J
(00736) J
(00737) J
(00738) J
(00739) J
(00740) J
(00741) J
(00742) J
(00743) J
(00744) J
(00745) J
(00746) J
(00747) J
(00748) J
(00749) J
(00750) J
(00751) J
(00752) J
(00753) J
(00754) J
(00755) J
(00756) J
(00757) J
(00758) J
(00759) J
(00760) J
(00761) J
(00762) J
(00763) J
(00764) J
(00765) J
(00766) J
(00767) J
(00768) J
(00769) J
(00770) J
(00771) J
(00772) J
(00773) J
(00774) J
(00775) J
(00776) J
(00777) J
(00778) J
(00779) J
(00780) J
(00781) J
(00782) J
(00783) J
(00784) J
(00785) J
(00786) J
(00787) J
(00788) J
(00789) J
(00790) J
(00791) J
(00792) J
(00793) J
(00794) J
(00795) J
(00796) J
(00797) J
(00798) J
(00799) J
(00800) J
(00801) J
(00802) J
(00803) J
(00804) J
(00805) J
(00806) J
(00807) J
(00808) J
(00809) J
(00810) J
(00811) J
(00812) J
(00813) J
(00814) J
(00815) J
(00816) J
(00817) J
(00818) J
(00819) J
(00820) J
(00821) J
(00822) J
(00823) J
(00824) J
(00825) J
(00826) J
(00827) J
(00828) J
(00829) J
(00830) J
(00831) J
(00832) J
(00833) J
(00834) J
(00835) J
(00836) J
(00837) J
(00838) J
(00839) J
(00840) J
(00841) J
(00842) J
(00843) J
(00844) J
(00845) J
(00846) J
(00847) J
(00848) J
(00849) J
(00850) J
(00851) J
(00852) J
(00853) J
(00854) J
(00855) J
(00856) J
(00857) J
(00858) J
(00859) J
(00860) J
(00861) J
(00862) J
(00863) J
(00864) J
(00865) J
(00866) J
(00867) J
(00868) J
(00869) J
(00870) J
(00871) J
(00872) J
(00873) J
(00874) J
(00875) J
(00876) J
(00877) J
(00878) J
(00879) J
(00880) J
(00881) J
(00882) J
(00883) J
(00884) J
(00885) J
(00886) J
(00887) J
(00888) J
(00889) J
(00890) J
(00891) J
(00892) J
(00893) J
(00894) J
(00895) J
(00896) J
(00897) J
(00898) J
(00899) J
(00900) J
(00901) J
(00902) J
(00903) J
(00904) J
(00905) J
(00906) J
(00907) J
(00908) J
(00909) J
(00910) J
(00911) J
(00912) J
(00913) J
(00914) J
(00915) J
(00916) J
(00917) J
(00918) J
(00919) J
(00920) J
(00921) J
(00922) J
(00923) J
(00924) J
(00925) J
(00926) J
(00927) J
(00928) J
(00929) J
(00930) J
(00931) J
(00932) J
(00933) J
(00934) J
(00935) J
(00936) J
(00937) J
(00938) J
(00939) J
(00940) J
(00941) J
(00942) J
(00943) J
(00944) J
(00945) J
(00946) J
(00947) J
(00948) J
(00949) J
(00950) J
(00951) J
(00952) J
(00953) J
(00954) J
(00955) J
(00956) J
(00957) J
(00958) J
(00959) J
(00960) J
(00961) J
(00962) J
(00963) J
(00964) J
(00965) J
(00966) J
(00967) J
(00968) J
(00969) J
(00970) J
(00971) J
(00972) J
(00973) J
(00974) J
(00975) J
(00976) J
(00977) J
(00978) J
(00979) J
(00980) J
(00981) J
(00982) J
(00983) J
(00984) J
(00985) J
(00986) J
(00987) J
(00988) J
(00989) J
(00990) J
(00991) J
(00992) J
(00993) J
(00994) J
(00995) J
(00996) J
(00997) J
(00998) J
(00999) J
(01000) J

```


PAGE 28: (BUND) <DCAL6>BUND16M.H50.2, 29-Dec-88 14:38:48, Eds WOLF
APU3-PITCH(V,A,U,B,C,D) COMPUTE AND CODE PITCH

(00768) * APU3-PITCH(V,A,U,B,C,D) COMPUTE AND CODE PITCH
(00769) * ALSO COMPUTE ONE-TAP COEFF
(00770) * AND OUTPUT 3 AUTOCORRELATION
(00771) * COEFF CENTERED AT THE PITCH LAG
(00772) *
(00773) *
(00774) *
(00775) *
(00776) *
(00777) *
(00778) *
(00779) *

INPUT: U AUTOCORRELATION COEFFICIENTS(U(14),...,U(133))
U

!!!! NOTE: BUFFER U MUST BE COMPACT-1111

CONSTANT = 14.0

OUTPUTS: PITCH LAG (IN SAMPLES)
SA CODED PITCH(INTEGER)
SB ONE-TAP COEFFICIENT
SD 3 AUTOCORRELATION COEFFICIENTS
Y CENTERED AT THE PITCH LAG

-U(PITCH LAG-1)
-U(PITCH LAG)
-U(PITCH LAG+1)

PROCEDURE:

DO SMAK(U) TO GET U(MAX) AND LOC OF MAX(=PITCH-14)
COMPUTE AND OUTPUT CODED PITCH=INTEGER OF MAX=>SB
COMPUTE AND OUTPUT PITCH LAG=MAX+14=>SA
COMPUTE INTEGER PITCH LAG FOR ADDRESS MOD
MODIFY APS ADDRESS
COMPUTE AND OUTPUT ONE-TAP COEFF =>SD
OUTPUT 3 AUTOCORRELATION COEFFS
CENTERED AT PITCH LAG ; U=>Y

INPUT STREAM: U(14),...,U(133), (FW1), (2**15), SC(CONST=14.0), (FW1),
, "MODIFIED INSTR", (FW1), U(0), U(PITCH LAG-1), U(PITCH LAG)
, U(PITCH LAG+1)

OUTPUT STREAM: SB(CODED PITCH), SA(PITCH LAG=MAX+14), "INTEGER
PITCH", "MODIFIED INSTR", SD(ONE TAP COEFF)
, 3 AUTOCORRELATION COEFF'S (Y(0), Y(1), Y(2)).

WHERE:

"INTEGER PITCH" IS USED BY APS OUTPUT PCM TO MODIFY
APS INPUT PCM
"MODIFIED INSTR" IS APS INSTR TO BE MODIFIED.
IT IS READ FROM BUS1, AND WRITTEN
TO APS PSEUDO MEMORY.

REGISTER USAGE:

SMAK SECTION:
ALL A REGISTERS ARE USED
LOC OF MAX STORED IN A3

(00780) *
(00781) *
(00782) *
(00783) *
(00784) *
(00785) *
(00786) *
(00787) *
(00788) *
(00789) *
(00790) *
(00791) *
(00792) *
(00793) *
(00794) *
(00795) *
(00796) *
(00797) *
(00798) *
(00799) *
(00800) *
(00801) *
(00802) *
(00803) *
(00804) *
(00805) *
(00806) *
(00807) *
(00808) *
(00809) *
(00810) *
(00811) *
(00812) *
(00813) *
(00814) *
(00815) *
(00816) *
(00817) *
(00818) *
(00819) *
(00820) *

PAGE 21: (00001)DCAL6200016M.MSD.2, 29-Dec-88 14:30:40, Ed: WOLF
APU3-PITCH(V,A,U,B,C,D) COMPUTE AND CODE PITCH

```

(00021) * NONE OF THE W REGISTERS ARE USED EXCEPT W7
(00022) * VALUE OF COEFF AT MAX LOC STORED IN W7
(00023) * OTHERWISE:
(00024) *
(00025) * A0 U(B)(SDIV) \ M0 SDIV \
(00026) * A1 SDIV, W(PITCH-1) \ W(PITCH) M1 2--15 \
(00027) * A2 U(MAX)/W(0) \ M2 --- \
(00028) * A3 LOC U(MAX) \ M3 --- \
(00029) * A4 CONST=14 \ M4 PITCH LAG=MAX+14, SDIV \
(00030) * A5 LAG=2--15 \ M5 LOC U(MAX)-1 \
(00031) * A6 SDIV, W(PITCH+1) \ M6 U(0)(SDIV) \
(00032) * A7 MAX=2--15 \ M7 U(MAX) \
(00033) *
(00034) * PITU - APW INITIALIZATION
(00035) *
(00036) * EVEN
(00037) * DATA PITU$A
(00038) * DATA PITU$Z
(00039) *
(00040) * PITU$1: BEGIN APW(PITU)
(00041) *
(00042) * DO SWAP ON V ARRAY (TERMINATE INPUT ON (PVI3))
(00043) *
(00044) * PITU$A1: NOP \ K(1) \ R=1
(00045) * MOV(IQA,A7) \ MOV(R,A4) \ A7=U(14) \ A4=1
(00046) * MOV(ZERO,A6) \ MOV(R,A6) \ A6=0 \ A6=1
(00047) * K(2) \ R=2
(00048) * NOP \ MOV(IQA,A7) \ A7=U(15)
(00049) * MOV(R,A5) \ MOV(I5),R(A7) \ AS=2 \ AS=2, R=U(15)
(00050) * JUMPS(PITU$1,PVI1)
(00051) *
(00052) * PITU$0: MOV(IQA,A0) \ NOP \ A0=UE
(00053) * MOV(A4),MAX(A7,A0) \ MOV(R,A7) \ A4=LOC, R=MAX \ A7=MAX
(00054) * CALLS(PITU$3,T2) \ IF NEW MAX
(00055) * NOP \ ADD(A5,A4) \ R=LOC
(00056) * JUMPS(PITU$4,PVI1) \ EXIT IF UBS IS 000
(00057) * MOV(R,A7) \ MOV(IQA,A0) \ A7=MAX \ A0=00
(00058) * CALLS(PITU$2,T1) \ IF NEW MAX \
(00059) * ADD(A5,A4) \ MOV(A4),MAX(A7,A0) \ R=LOC \ A4=LOC, R=MAX
(00060) * JUMPS(PITU$0,PVI1) \ EXIT IF UBS IS EVEN
(00061) *
(00062) *
(00063) * PITU$1: R(A7) \ NOP \ EXIT IF LOOP BOTTOM
(00064) * MOV(R,A1) \ MOV(R,RX0) \ R=MAX
(00065) * CALLS(PITU$3,T2) \ A1=MAX \ EXO=MAX
(00066) * MOV(RX1,A2) \ NOP \ A2=MAX \ IF NEW MAX
(00067) * JUMPS(PITU$5) \ TO COMMON CLEAN UP
(00068) *
(00069) *
(00070) * PITU$2: R(A4) \ NOP \ SUBROUTINE FOR LM SIDE
(00071) * MOV(R,A6) \ NOP \ R=LOC
(00072) * RETURN \ A6=LOC
(00073) *

```

PAGE 22: (00001)OCAL16)BOMICH.N50.2, 29-Dec-88 14:30:00, R01 WOLF
AP03-PITCH(Y,A,B,C,B)
COMPUTE AND CODE PITCH

```

(00074) *
A10 05F14 00000200 PITUS3: MOP \ R(A4)
A11 05F16 00000096 MOP \ MOV(R,A6)
A12 05F18 00001000 RETURN
(00075) *
A13 05F1A 00000200 PITUS4: MOP \ R(A7)
A14 05F1C 00000098 MOP(R,A1) \ MOV(R,EXO)
A15 05F1E 00000015 CALLS(PITUS2,T1)
A16 05F20 00000000 MOV(EXT,A2) \ MOP
(00076) *
A17 05F22 00000200 PITUS5: MAX(A1,A2) \ R(A6)
A18 05F24 00000000 MOP(R,M7) \ MOP
A19 05F26 00000025 JUMPS(PITUS6,T1)
A20 05F28 00000000 R(A6) \ MOP
A21 05F2A 00000000 MOP(R,A3) \ MOP
A22 05F2C 00000027 JUMP(PITUS7)
A23 05F2E 00000098 MOP \ MOV(R,EXO)
A24 05F30 00000000 MOV(EXT,A3) \ MOP
(00077) *
A25 05F32 00000200 PITUS7: CLEAR(M1)
A26 05F34 00000000 MOV(LOI,M1) \ MOP
A27 05F36 00000000 R(A3) \ MOP
A28 05F38 00000000 MOP(R,M5) \ MOP
A29 05F3A 00000000 MOP(M1,M5) \ MOP
A30 05F3C 00000000 MOV(P,A7) \ MOP
A31 05F3E 00000000 ALIGN(A7) \ MOP
(00078) *
A32 05F40 00000000 MOV(R,OQ) \ MOP
(00079) *
A33 05F42 00000000 ADD 14 TO MAX LOC TO GET PITCH LAG (FLT POINT VALUE)
A34 05F44 00000000 MOV(LOI,A4) \ MOP
A35 05F46 00000000 ADD(A3,A4) \ MOP
(00080) *
A36 05F48 00000000 MOV(R,OQ) \ MOP
(00081) *
A37 05F4A 00000000 GENERATE INTEGER PITCH LAG FOR ADDRESS MODIFICATION
A38 05F4C 00000000 (MULT BY (2**15), THEN ALIGN)
A39 05F4E 00000000 MOV(R,M4) \ MOP
A40 05F50 00000000 MOP(M1,M4) \ MOP
A41 05F52 00000000 MOV(P,A5) \ MOP
A42 05F54 00000000 ALIGN(A5) \ MOP
(00082) *
A43 05F56 00000000 MOV(R,M4) \ MOP
A44 05F58 00000000 MOP(M1,M4) \ MOP
A45 05F5A 00000000 MOV(P,A5) \ MOP
A46 05F5C 00000000 ALIGN(A5) \ MOP
(00083) *
A47 05F5E 00000000 SEND TO BUS1 TO MODIFY BUS1 COPY OF APS INSTR.
(00084) *
A48 05F60 00000000
(00085) *
A49 05F62 00000000
(00086) *
A50 05F64 00000000
(00087) *
A51 05F66 00000000
(00088) *
A52 05F68 00000000
(00089) *
A53 05F6A 00000000
(00090) *
A54 05F6C 00000000
(00091) *
A55 05F6E 00000000
(00092) *
A56 05F70 00000000
(00093) *
A57 05F72 00000000
(00094) *
A58 05F74 00000000
(00095) *
A59 05F76 00000000
(00096) *
A60 05F78 00000000
(00097) *
A61 05F7A 00000000
(00098) *
A62 05F7C 00000000
(00099) *
A63 05F7E 00000000
(00100) *
A64 05F80 00000000
(00101) *
A65 05F82 00000000
(00102) *
A66 05F84 00000000
(00103) *
A67 05F86 00000000
(00104) *
A68 05F88 00000000
(00105) *
A69 05F8A 00000000
(00106) *
A70 05F8C 00000000
(00107) *
A71 05F8E 00000000
(00108) *
A72 05F90 00000000
(00109) *
A73 05F92 00000000
(00110) *
A74 05F94 00000000
(00111) *
A75 05F96 00000000
(00112) *
A76 05F98 00000000
(00113) *
A77 05F9A 00000000
(00114) *
A78 05F9C 00000000
(00115) *
A79 05F9E 00000000
(00116) *
A80 05FA0 00000000
(00117) *
A81 05FA2 00000000
(00118) *
A82 05FA4 00000000
(00119) *
A83 05FA6 00000000
(00120) *
A84 05FA8 00000000
(00121) *
A85 05FAA 00000000
(00122) *
A86 05FAC 00000000
(00123) *
A87 05FAE 00000000
(00124) *
A88 05FB0 00000000
(00125) *
A89 05FB2 00000000
(00126) *
A90 05FB4 00000000
(00127) *
A91 05FB6 00000000
(00128) *
A92 05FB8 00000000
(00129) *
A93 05FBA 00000000
(00130) *
A94 05FBC 00000000
(00131) *
A95 05FBE 00000000
(00132) *
A96 05FB0 00000000
(00133) *
A97 05FB2 00000000
(00134) *
A98 05FB4 00000000
(00135) *
A99 05FB6 00000000
(00136) *
A100 05FB8 00000000
(00137) *
A101 05FBA 00000000
(00138) *
A102 05FBC 00000000
(00139) *
A103 05FBE 00000000
(00140) *
A104 05FB0 00000000
(00141) *
A105 05FB2 00000000
(00142) *
A106 05FB4 00000000
(00143) *
A107 05FB6 00000000
(00144) *
A108 05FB8 00000000
(00145) *
A109 05FBA 00000000
(00146) *
A110 05FBC 00000000
(00147) *
A111 05FBE 00000000
(00148) *
A112 05FB0 00000000
(00149) *
A113 05FB2 00000000
(00150) *
A114 05FB4 00000000
(00151) *
A115 05FB6 00000000
(00152) *
A116 05FB8 00000000
(00153) *
A117 05FBA 00000000
(00154) *
A118 05FBC 00000000
(00155) *
A119 05FBE 00000000
(00156) *
A120 05FB0 00000000
(00157) *
A121 05FB2 00000000
(00158) *
A122 05FB4 00000000
(00159) *
A123 05FB6 00000000
(00160) *
A124 05FB8 00000000
(00161) *
A125 05FBA 00000000
(00162) *
A126 05FBC 00000000
(00163) *
A127 05FBE 00000000
(00164) *
A128 05FB0 00000000
(00165) *
A129 05FB2 00000000
(00166) *
A130 05FB4 00000000
(00167) *
A131 05FB6 00000000
(00168) *
A132 05FB8 00000000
(00169) *
A133 05FBA 00000000
(00170) *
A134 05FBC 00000000
(00171) *
A135 05FBE 00000000
(00172) *
A136 05FB0 00000000
(00173) *
A137 05FB2 00000000
(00174) *
A138 05FB4 00000000
(00175) *
A139 05FB6 00000000
(00176) *
A140 05FB8 00000000
(00177) *
A141 05FBA 00000000
(00178) *
A142 05FBC 00000000
(00179) *
A143 05FBE 00000000
(00180) *
A144 05FB0 00000000
(00181) *
A145 05FB2 00000000
(00182) *
A146 05FB4 00000000
(00183) *
A147 05FB6 00000000
(00184) *
A148 05FB8 00000000
(00185) *
A149 05FBA 00000000
(00186) *
A150 05FBC 00000000
(00187) *
A151 05FBE 00000000
(00188) *
A152 05FB0 00000000
(00189) *
A153 05FB2 00000000
(00190) *
A154 05FB4 00000000
(00191) *
A155 05FB6 00000000
(00192) *
A156 05FB8 00000000
(00193) *
A157 05FBA 00000000
(00194) *
A158 05FBC 00000000
(00195) *
A159 05FBE 00000000
(00196) *
A160 05FB0 00000000
(00197) *
A161 05FB2 00000000
(00198) *
A162 05FB4 00000000
(00199) *
A163 05FB6 00000000
(00200) *
A164 05FB8 00000000
(00201) *
A165 05FBA 00000000
(00202) *
A166 05FBC 00000000
(00203) *
A167 05FBE 00000000
(00204) *
A168 05FB0 00000000
(00205) *
A169 05FB2 00000000
(00206) *
A170 05FB4 00000000
(00207) *
A171 05FB6 00000000
(00208) *
A172 05FB8 00000000
(00209) *
A173 05FBA 00000000
(00210) *
A174 05FBC 00000000
(00211) *
A175 05FBE 00000000
(00212) *
A176 05FB0 00000000
(00213) *
A177 05FB2 00000000
(00214) *
A178 05FB4 00000000
(00215) *
A179 05FB6 00000000
(00216) *
A180 05FB8 00000000
(00217) *
A181 05FBA 00000000
(00218) *
A182 05FBC 00000000
(00219) *
A183 05FBE 00000000
(00220) *
A184 05FB0 00000000
(00221) *
A185 05FB2 00000000
(00222) *
A186 05FB4 00000000
(00223) *
A187 05FB6 00000000
(00224) *
A188 05FB8 00000000
(00225) *
A189 05FBA 00000000
(00226) *
A190 05FBC 00000000
(00227) *
A191 05FBE 00000000
(00228) *
A192 05FB0 00000000
(00229) *
A193 05FB2 00000000
(00230) *
A194 05FB4 00000000
(00231) *
A195 05FB6 00000000
(00232) *
A196 05FB8 00000000
(00233) *
A197 05FBA 00000000
(00234) *
A198 05FBC 00000000
(00235) *
A199 05FBE 00000000
(00236) *
A200 05FB0 00000000
(00237) *
A201 05FB2 00000000
(00238) *
A202 05FB4 00000000
(00239) *
A203 05FB6 00000000
(00240) *
A204 05FB8 00000000
(00241) *
A205 05FBA 00000000
(00242) *
A206 05FBC 00000000
(00243) *
A207 05FBE 00000000
(00244) *
A208 05FB0 00000000
(00245) *
A209 05FB2 00000000
(00246) *
A210 05FB4 00000000
(00247) *
A211 05FB6 00000000
(00248) *
A212 05FB8 00000000
(00249) *
A213 05FBA 00000000
(00250) *
A214 05FBC 00000000
(00251) *
A215 05FBE 00000000
(00252) *
A216 05FB0 00000000
(00253) *
A217 05FB2 00000000
(00254) *
A218 05FB4 00000000
(00255) *
A219 05FB6 00000000
(00256) *
A220 05FB8 00000000
(00257) *
A221 05FBA 00000000
(00258) *
A222 05FBC 00000000
(00259) *
A223 05FBE 00000000
(00260) *
A224 05FB0 00000000
(00261) *
A225 05FB2 00000000
(00262) *
A226 05FB4 00000000
(00263) *
A227 05FB6 00000000
(00264) *
A228 05FB8 00000000
(00265) *
A229 05FBA 00000000
(00266) *
A230 05FBC 00000000
(00267) *
A231 05FBE 00000000
(00268) *
A232 05FB0 00000000
(00269) *
A233 05FB2 00000000
(00270) *
A234 05FB4 00000000
(00271) *
A235 05FB6 00000000
(00272) *
A236 05FB8 00000000
(00273) *
A237 05FBA 00000000
(00274) *
A238 05FBC 00000000
(00275) *
A239 05FBE 00000000
(00276) *
A240 05FB0 00000000
(00277) *
A241 05FB2 00000000
(00278) *
A242 05FB4 00000000
(00279) *
A243 05FB6 00000000
(00280) *
A244 05FB8 00000000
(00281) *
A245 05FBA 00000000
(00282) *
A246 05FBC 00000000
(00283) *
A247 05FBE 00000000
(00284) *
A248 05FB0 00000000
(00285) *
A249 05FB2 00000000
(00286) *
A250 05FB4 00000000
(00287) *
A251 05FB6 00000000
(00288) *
A252 05FB8 00000000
(00289) *
A253 05FBA 00000000
(00290) *
A254 05FBC 00000000
(00291) *
A255 05FBE 00000000
(00292) *
A256 05FB0 00000000
(00293) *
A257 05FB2 00000000
(00294) *
A258 05FB4 00000000
(00295) *
A259 05FB6 00000000
(00296) *
A260 05FB8 00000000
(00297) *
A261 05FBA 00000000
(00298) *
A262 05FBC 00000000
(00299) *
A263 05FBE 00000000
(00300) *
A264 05FB0 00000000
(00301) *
A265 05FB2 00000000
(00302) *
A266 05FB4 00000000
(00303) *
A267 05FB6 00000000
(00304) *
A268 05FB8 00000000
(00305) *
A269 05FBA 00000000
(00306) *
A270 05FBC 00000000
(00307) *
A271 05FBE 00000000
(00308) *
A272 05FB0 00000000
(00309) *
A273 05FB2 00000000
(00310) *
A274 05FB4 00000000
(00311) *
A275 05FB6 00000000
(00312) *
A276 05FB8 00000000
(00313) *
A277 05FBA 00000000
(00314) *
A278 05FBC 00000000
(00315) *
A279 05FBE 00000000
(00316) *
A280 05FB0 00000000
(00317) *
A281 05FB2 00000000
(00318) *
A282 05FB4 00000000
(00319) *
A283 05FB6 00000000
(00320) *
A284 05FB8 00000000
(00321) *
A285 05FBA 00000000
(00322) *
A286 05FBC 00000000
(00323) *
A287 05FBE 00000000
(00324) *
A288 05FB0 00000000
(00325) *
A289 05FB2 00000000
(00326) *
A290 05FB4 00000000
(00327) *
A291 05FB6 00000000
(00328) *
A292 05FB8 00000000
(00329) *
A293 05FBA 00000000
(00330) *
A294 05FBC 00000000
(00331) *
A295 05FBE 00000000
(00332) *
A296 05FB0 00000000
(00333) *
A297 05FB2 00000000
(00334) *
A298 05FB4 00000000
(00335) *
A299 05FB6 00000000
(00336) *
A300 05FB8 00000000
(00337) *
A301 05FBA 00000000
(00338) *
A302 05FBC 00000000
(00339) *
A303 05FBE 00000000
(00340) *
A304 05FB0 00000000
(00341) *
A305 05FB2 00000000
(00342) *
A306 05FB4 00000000
(00343) *
A307 05FB6 00000000
(00344) *
A308 05FB8 00000000
(00345) *
A309 05FBA 00000000
(00346) *
A310 05FBC 00000000
(00347) *
A311 05FBE 00000000
(00348) *
A312 05FB0 00000000
(00349) *
A313 05FB2 00000000
(00350) *
A314 05FB4 00000000
(00351) *
A315 05FB6 00000000
(00352) *
A316 05FB8 00000000
(00353) *
A317 05FBA 00000000
(00354) *
A318 05FBC 00000000
(00355) *
A319 05FBE 00000000
(00356) *
A320 05FB0 00000000
(00357) *
A321 05FB2 00000000
(00358) *
A322 05FB4 00000000
(00359) *
A323 05FB6 00000000
(00360) *
A324 05FB8 00000000
(00361) *
A325 05FBA 00000000
(00362) *
A326 05FBC 00000000
(00363) *
A327 05FBE 00000000
(00364) *
A328 05FB0 00000000
(00365) *
A329 05FB2 00000000
(00366) *
A330 05FB4 00000000
(00367) *
A331 05FB6 00000000
(00368) *
A332 05FB8 00000000
(00369) *
A333 05FBA 00000000
(00370) *
A334 05FBC 00000000
(00371) *
A335 05FBE 00000000
(00372) *
A336 05FB0 00000000
(00373) *
A337 05FB2 00000000
(00374) *
A338 05FB4 00000000
(00375) *
A339 05FB6 00000000
(00376) *
A340 05FB8 00000000
(00377) *
A341 05FBA 00000000
(00378) *
A342 05FBC 00000000
(00379) *
A343 05FBE 00000000
(00380) *
A344 05FB0 00000000
(00381) *
A345 05FB2 00000000
(00382) *
A346 05FB4 00000000
(00383) *
A347 05FB6 00000000
(00384) *
A348 05FB8 00000000
(00385) *
A349 05FBA 00000000
(00386) *
A350 05FBC 00000000
(00387) *
A351 05FBE 00000000
(00388) *
A352 05FB0 00000000
(00389) *
A353 05FB2 00000000
(00390) *
A354 05FB4 00000000
(00391) *
A355 05FB6 00000000
(00392) *
A356 05FB8 00000000
(00393) *
A357 05FBA 00000000
(00394) *
A358 05FBC 00000000
(00395) *
A359 05FBE 00000000
(00396) *
A360 05FB0 00000000
(00397) *
A361 05FB2 00000000
(00398) *
A362 05FB4 00000000
(00399) *
A363 05FB6 00000000
(00400) *
A364 05FB8 00000000
(00401) *
A365 05FBA 00000000
(00402) *
A366 05FBC 00000000
(00403) *
A367 05FBE 00000000
(00404) *
A368 05FB0 00000000
(00405) *
A369 05FB2 00000000
(00406) *
A370 05FB4 00000000
(00407) *
A371 05FB6 00000000
(00408) *
A372 05FB8 00000000
(00409) *
A373 05FBA 00000000
(00410) *
A374 05FBC 00000000
(00411) *
A375 05FBE 00000000
(00412) *
A376 05FB0 00000000
(00413) *
A377 05FB2 00000000
(00414) *
A378 05FB4 00000000
(00415) *
A379 05FB6 00000000
(00416) *
A380 05FB8 00000000
(00417) *
A381 05FBA 00000000
(00418) *
A382 05FBC 00000000
(00419) *
A383 05FBE 00000000
(00420) *
A384 05FB0 00000000
(00421) *
A385 05FB2 00000000
(00422) *
A386 05FB4 00000000
(00423) *
A387 05FB6 00000000
(00424) *
A388 05FB8 00000000
(00425) *
A389 05FBA 00000000
(00426) *
A390 05FBC 00000000
(00427) *
A391 05FBE 00000000
(00428) *
A392 05FB0 00000000
(00429) *
A393 05FB2 00000000
(00430) *
A394 05FB4 00000000
(00431) *
A395 05FB6 00000000
(00432) *
A396 05FB8 00000000
(00433) *
A397 05FBA 00000000
(00434) *
A398 05FBC 00000000
(00435) *
A399 05FBE 00000000
(00436) *
A400 05FB0 00000000
(00437) *
A401 05FB2 00000000
(00438) *
A402 05FB4 00000000
(00439) *
A403 05FB6 00000000
(00440) *
A404 05FB8 00000000
(00441) *
A405 05FBA 00000000
(00442) *
A406 05FBC 00000000
(00443) *
A407 05FBE 00000000
(00444) *
A408 05FB0 00000000
(00445) *
A409 05FB2 00000000
(00446) *
A410 05FB4 00000000
(00447) *
A411 05FB6 00000000
(00448) *
A412 05FB8 00000000
(00449) *
A413 05FBA 00000000
(00450) *
A414 05FBC 00000000
(00451) *
A415 05FBE 00000000
(00452) *
A416 05FB0 00000000
(00453) *
A417 05FB2 00000000
(00454) *
A418 05FB4 00000000
(00455) *
A419 05FB6 00000000
(00456) *
A420 05FB8 00000000
(00457) *
A421 05FBA 00000000
(00458) *
A422 05FBC 00000000
(00459) *
A423 05FBE 00000000
(00460) *
A424 05FB0 00000000
(00461) *
A425 05FB2 00000000
(00462) *
A426 05FB4 00000000
(00463) *
A427 05FB6 00000000
(00464) *
A428 05FB8 00000000
(00465) *
A429 05FBA 00000000
(00466) *
A430 05FBC 00000000
(00467) *
A431 05FBE 00000000
(00468) *
A432 05FB0 00000000
(00469) *
A433 05FB2 00000000
(00470) *
A434 05FB4 00000000
(00471) *
A435 05FB6 00000000
(00472) *
A436 05FB8 00000000
(00473) *
A437 05FBA 00000000
(00474) *
A438 05FBC 00000000
(00475) *
A439 05FBE 00000000
(00476) *
A440 05FB0 00000000
(00477) *
A441 05FB2 00000000
(00478) *
A442 05FB4 00000000
(00479) *
A443 05FB6 00000000
(00480) *
A444 05FB8 
```


PAGE 231 (000010C0A1000010M.N00.2, 29-Dec-00 14:30:40, Ed: WOLF
AP03-PITCH(A,B,C,D) COMPUTE AND CODE PITCH

```

A36 05F50 00C0000 (00927) * MOV(R,00) \ NOP      ; 00 <= INTEGER PITCH
(00928) *
(00929) * WAIT TIL BUS1 COPY IS MODIFIED, THEN READ
(00930) * IT AND WRITE TO APS PSEUDO MEMORY.
(00931) * (ALSO CLEAR WE SO APS INPUT PCN WILL CONTINUE.)
(00932) * (IT WAS WAITING FOR WRITE TO COMPLETE.)
(00933) *
A37 05F52 9010037 (00934) * PITCHW1: JUMPC(PITCHW1,00E)  ; LOOP WHILE OUTPUT QUEUE NOT EMPTY
A38 05F54 0000000 (00935) * NOP                      ; DELAY 1 FOR MEMORY WRITE
A39 05F56 20372037 (00936) * CLEAR(W1)                ; NOW LET APS PROCEED
(00937) *
A3A 05F58 00C0000 (00938) * MOV(10A,00) \ NOP      ; THEN READ MODIFIED APS INSTR
(00939) *
(00940) *
(00941) * WAIT TIL APS COPY IS WRITTEN , THEN PROCEED APS INPUT PCN AGAIN
(00942) *
A3B 05F5A 901003B (00943) * PITCHW2: JUMPC(PITCHW2,00E)  ; LOOP WHILE OUTPUT QUEUE NOT EMPTY
A3C 05F5C 0000000 (00944) * NOP                      ; DELAY 1 AS ABOVE
(00945) *
A3D 05F5E 20372037 (00946) * CLEAR(W1)
(00947) *
(00948) * SET UP DECS FOR DIVIDE: 0(MAX)/0(0)
(00949) * M7 ALREADY = 0(MAX)
(00950) * GET 0(0) FROM IO
(00951) *
A3E 05F60 00C7000 (00952) * MOV(10A,M6) \ NOP      M6 = 0(0) \ ---
A3F 05F62 00F0000 (00953) * MOV(10A,A6) \ NOP      A6 = 0(0) \ ---
(00954) *
(00955) * DO SOIV: DIVIDE 0/C
(00956) * EXPECTS (IN LEFT ADM) 0 IN M7
(00957) * C IN M6
(00958) * C IN A6
(00959) *
PITCH0: K(2) \ NOP      R=2
MOV(R,A6) \ NOP      A6=2
RCP(A6) \ NOP      R1=1/C+DEL(=F0)(= 1/C + 6)
MOV(R,M6) \ NOP      M6=F0
MUL(M6,M6) \ NOP      P1=F0^2
MOV(P,A1) \ NOP      A1=F0^2
SUB(A6,A1) \ NOP      R2=2-F0^2 (= 1 - C6)
MOV(R,M4) \ NOP      M4=R2
MUL(M6,M4) \ NOP      P2=F0^2*R2(=P1)(= 1/C - (6^2)C)
MOV(P,M6) \ NOP      M6=P1
MUL(M6,M6) \ NOP      P3=P1^2
SUB(A6,A1) \ NOP      R3=2-P1^2
MOV(R,M4) \ NOP      M4=R3
MUL(M6,M4) \ NOP      P4=P1*(2-P1^2)(=P2)(= 1/C - (6^2)C(C^2)) (=1/C)
MOV(P,M6) \ NOP      M6=P2
MUL(M6,M7) \ NOP      P5=P2^2 (=R/C)
MOV(P,A2) \ NOP      A2 = TAP \ ---
(00970) *
(00971) * SOIV COMPLETE - NEGATE ONE-TAP COEFF AND OUTPUT
(00972) *
(00973) *
(00974) *
(00975) *
(00976) *
(00977) *
(00978) *
(00979) *

```

```

PAGE 24: (00007)BCA16>00016M.N50.2, 29-0ac-88 14130148, Ed: WOLF
          APW3-PITCH(V,A,W,C,D)
          *
          (00008) *
A52 05700 01400000      WEC(A2)\NOP      JN--W(MAX)/W(0)
A53 0570A 009C0000      MOV(R,DQ)\NOP    JOUTPUT NEGATED ONE-YAR CORRP
          *
          (00003) *
          (00004) *
          (00005) *
          *
          INPUT (AND OUTPUT) CORRPS CENTERED AT PITCH LAG
          *
          (00006) *
          (00007) *
A54 0570C 00F10000      MOV(IQA,A1)\NOP    JAL-W(PITCH-1)\
A55 0570E 012000F1      WEC(A1)\MOV(IQA,A1)  JN--W(PITCH-1)\AL-W(PITCH)
A56 05708 00F40A20      MOV(IQA,A6)\WEC(A1)  JAG-W(PITCH+1)\W--W(PITCH)
A57 05792 009C0000      MOV(R,DQ)\NOP        JF(0)--W(PITCH-1)\
A58 05794 01C0009C      WEC(A6)\MOV(R,DQ)    JN--W(PITCH+1)\W(1)--W(PITCH)
A59 05796 009C0000      MOV(R,DQ)\NOP        JF(2)--W(PITCH+1)
          *
          (00009) *
          (00004) *
A5A 05798 20322032      CLEAR(RA)
A5B 0579A 10000000      JUMP(0)
          *
          (00007) *
          *
          PITCH$Z=BA-PITCH$A
          END
          0579C
          (00005)

```


PAGE 26: (00001)<0CA16>000164.MSD.2, 29-Dec-88 14:38:48, ED: WOLF
IP53-PTC00(V,A,U,B,C,D)

ADDRESS	OPERATION	COMMENT	MODULE SIZE
(01159) *			
(01160) *	GEN D SCALAR ADDR		
(01161) *			
(01162) *	MOVW(BW3,BW3,TF)		
(01163) *			
(01164) *	GENERATE AUTOCORRELATION CORRP ADDR		
(01165) *			
(01166) *	LOAD(BW6,C0,TF)		
(01167) *	LOAD(BW2,M\$)		
(01168) *	ADD(BW6,M\$5,TF)		
(01169) *	ADD(BW6,C0,TF)		
(01170) *			
(01171) *	CLEAR(R0)		
(01172) *	HOP(0)		
(01173) *			
(01174) *	P1TS\$A=BC		
(01175) *			
(01176) *	END		
(01177) *			
(01178) *	P1TS\$I DATA 6P*0.0*		
...			
(01179) *	P1TS\$2=BL-P1TS\$		

PAGE 29: [0000]C0CA16>00016M.MSD.2, 29-Dec-80 14:30:40, Ed: WOLF
 MWLQ6(Y,A,U,V,W) WEINER-LEVINSON MATRIX SOL'N & P-6 CODING/QUANT.

MWLQ6(Y,A,U,V,W) WEINER-LEVINSON MATRIX SOL'N & P-6 CODING/QUANT.

MWLQ6-APU PROGRAM

THE CHANGES FROM MWLQ IN BN300-MSO ARE:

1. CODE/QUANTIZE 6 COEFFICIENTS INSTEAD OF 8.
2. THE QUANTIZED E-VALUES ARE REGULAR FLY PT, NOT 16-BIT FIXED.
3. THEREFORE THE THRESH/VALUE TABLES ARE FLOATING. SEPARATE, NOT INTERLEAVED, THRESH/VALUE TABLES ARE USED. THE VALUE TABLE IS THE K-DECODING TABLE.

WEINER LEVINSON DURHAM INVERSE (MWLF) FOLLOWED BY 6TH ORDER REFLECTION COEFFICIENT CODING AND QUANTIZATION (EQWAN)

MATHEMATICS

SEE J. MARKOWL, LINEAR PREDICTION REVIEW
 PROC. IEEE, VOL 63, PPS66, APR 1975

BUFFER DEFINITIONS FOR THE MWLF PART:

V(K), AUTOCORRELATION COEFFICIENTS, $V(K)=E(K)$
 V(K) MUST BE COMPACT 32 BIT FLOATING POINT
 VSIZE NOT USED IN COMPUTATION, BUT
 VSIZE > USIZE FOR VALID RESULTS.

U(K), A(K) COEFFICIENTS
 U(K) MUST BE COMPACT 32 BIT FLOATING POINT
 U(K)=A(K-1), 16.A(0)=1 NOT IN BUFFER

V(K), PARTIAL CORRELATIONS AND ERROR
 V(K) MUST BE COMPACT 32 BIT FLOATING POINT
 VSIZE=2*USIZE
 V(K)=P(K-1), 0<K<USIZE
 V(K*USIZE)=E(K-1), USIZE<K<VSIZE

A, STABILITY TEST PARAMETER
 IF P(N)> CONTENTS OF (A), THEN
 FOR J>0
 A(N+J)=P(N+J)=0, E(N+J)=E(N)

FOR THE EQWAN PART, OUTPUT BUFFERS:
 THE W(K) BUFFER IS OVERWRITTEN BY THE QUANTIZED REFLECTION
 (PARTIAL CORRELATION) COEFFICIENTS.

W(K) = 6 CODED REFLECTION COEFFICIENTS. W IS A COMPACT LONG (16 BIT) FIXED POINT BUFFER.

MWLQ-APU INITIALIZATION

PAGE 301 (888D3C0C16)B0H16M, NS0.2, 29-Dec-88 14:30:40, Ed: WOLF
 WMLQ(V,A,U,V,U) WEINER-LEVINSON MATRIX SOL-M & P-6 CODING/QUANT.

		EVEN	DATA	WMLFSA	WMLFSSZ	START ON WORD BOUNDARY	START ADDRESS	SIZE
06014 0000	(01233)							
06015 0065	(01234)							
	(01235)							
	(01236)							
	(01237)							
	(01238)							
	(01239)							
	(01240)							
A00 06016 00P70000	(01241)	WMLFSA	MOV(IQA,A7)\NOP					
A01 06018 00C000C	(01242)		MOV(ZERO,M4)					
A02 0601A 00C0000	(01243)		MOV(IQ,M5)\NOP					
A03 0601C 00F50000	(01244)		MOV(IQA,A5)\NOP					
	(01245)							
	(01246)							
	(01247)							
	(01248)							
	(01249)							
	(01250)							
A04 0601E 2EA02E00	(01251)		RCP(A5)					
A05 06020 00P70000	(01252)		MOV(IQA,A2)\NOP					
A06 06022 0000000	(01253)		MOV(ZERO,M0)					
A07 06024 04000400	(01254)		MUL(M0,M4)					
A08 06026 02490249	(01255)		MOV(M1), R(A2)					
A09 06028 900A004F	(01256)		JUMPC(WMLFSA,AF2)					
	(01257)							
APA 0602A 90400010	(01258)		JUMPC(WMLFSE,AF3),SET					
	(01259)							
A0B 0602C 20202020	(01260)		CLEAR(AF3)					
	(01261)							
A0C 0602E 08EA0000	(01262)		MOV(IQA,M2)\NOP					
A0D 06030 00EF0000	(01263)		MOV(IQA,M6)\NOP					
A0E 06032 05500550	(01264)		MOV(A0), MUL(M2,M6)					
A0F 06034 42124212	(01265)		MOV(A2), ADD(A0,A2)					
	(01266)							
A10 06036 08E00000	(01267)	WMLFSE	MOV(IQA,M3)\NOP					
A11 06038 00EF0000	(01268)		MOV(IQA,M7)\NOP					
A12 0603A 05P105P1	(01269)		MOV(A1), MUL(M3,M7)					
A13 0603C 42242232	(01270)		MOV(A2), ADD(A1,A2)					
	(01271)							
A14 0603E 9016000C	(01272)		JUMPC(02,FV1)					
	(01273)							
A15 06040 00000000	(01274)		MOV(P,A0)					
A16 06042 42124212	(01275)		MOV(A2), ADD(A0,A2)					
	(01276)							
A17 06044 20372037	(01277)	WMLFRE	CLEAR(M1)					
	(01278)							
	(01279)							
	(01280)							
	(01281)							
	(01282)							
	(01283)							
	(01284)							
	(01285)							

(01200)	00000	00000	MUL(M1,M5)	E* RCP(E)
(01201)	00000	00000	MOV(M2), K(2)	M2=S(M)
(01202)	00000	00000	MOV(P,A0)	A0=E* RCP(E)
(01203)	00000	00000	MOV(A1), SUB(A1,A0)	A1=2
(01204)	00000	00000	MOV(M6), K(1)	M6=2-E* RCP(E)
(01205)	00000	00000	MUL(M1,M6)	
(01206)	00000	00000	MOV(P,M1)	M1=A4=EIR,
(01207)	00000	00000	MOV(A4), MUL(M1,M5)	EIR APPROX(1/E) TO 2*(-11)
(01208)	00000	00000	MOV(P,A0)	
(01209)	00000	00000	MOV(A0), SUB(A6,A0)	A6=1
(01210)	00000	00000	MOV(R,M6)	M6=1-EIR* E
(01211)	00000	00000	MUL(M1,M6)	
(01212)	00000	00000	MOV(P,A0)	A0=EIR(1-E* EIR)
(01213)	00000	00000	ADD(A0,A4)	
(01214)	00000	00000	MOV(R,M6)	R=1/(S(M-1), FULL PRECISION
(01215)	00000	00000	MUL(M2,M6)	
(01216)	00000	00000	MOV(P,A4)	
(01217)	00000	00000	R(A4)	
(01218)	00000	00000	MOV(P,M0)	M0=M4-A4=P(M)
(01219)	00000	00000	MOV(P,M4)	Q0 = P(M) (FIXED) = A(M,M)
(01220)	00000	00000	MOV(R,Q0)\MOP	
(01300)	00000	00000		
(01301)	00000	00000		
(01302)	00000	00000		
(01303)	00000	00000		
(01304)	00000	00000		
(01305)	00000	00000		
(01306)	00000	00000		
(01307)	00000	00000		
(01308)	00000	00000		
(01309)	00000	00000		
(01310)	00000	00000		
(01311)	00000	00000		
(01312)	00000	00000		
(01313)	00000	00000		
(01314)	00000	00000		
(01315)	00000	00000		
(01316)	00000	00000		
(01317)	00000	00000		
(01318)	00000	00000		
(01319)	00000	00000		
(01320)	00000	00000		
(01321)	00000	00000		
(01322)	00000	00000		
(01323)	00000	00000		
(01324)	00000	00000		
(01325)	00000	00000		
(01326)	00000	00000		
(01327)	00000	00000		
(01328)	00000	00000		
(01329)	00000	00000		
(01330)	00000	00000		
(01331)	00000	00000		
(01332)	00000	00000		
(01333)	00000	00000		
(01334)	00000	00000		
(01335)	00000	00000		
(01336)	00000	00000		
(01337)	00000	00000		
(01338)	00000	00000		

PAGE 33: CMMND3<DCAL16>BWM16M.NSO.2, 29-Dec-88 14:38:48, E4: WOLF
WMLQ6(Y,L,U,V,M) WEINER-LEVINSON MATRIX SOL'N & P-6 CODING/QUANT.

```

A37 06004 00000000 (01339) NOP
A38 06006 9016002F (01340) JUMPC(03,FH1)
A39 06008 20372037 (01341) *
A39 06008 20372037 (01342) WMLFAE CLEAR(VI)
A3A 0600A 04110411 (01343) MOV(A1), MUL(M0,M4)
A3B 0600C 433C0000 (01344) MOV(OQ), ADD(A1,A3)\NOP
(01345) *
(01346) *
(01347) * WMLF-APU CALCULATION OF E(M)
(01348) * E(M)=C1-P(M)P(M)E(M-1)
(01349) *
(01350) * REGISTER CONTENTS AT START
(01351) *
(01352) * M0=M4=A4=P(M), M5=A5=E(M-1), A6=1, A7=TEST
(01353) * P=P(M)P(M), R=P(M)=A(M,M)
(01354) * DATA SEQUENCE
(01355) * OTPT P(M), E(M)
A3C 0600E 00000000 (01356) AB=P(M)P(M)
A3D 0600F 40DC0000 (01357) OQ=A(M,M)
A3E 06012 020A020A (01358) M2=1-P(M)P(M)
A3F 06014 05200520 (01359) E(M-1)=C1-P(M)P(M)
A40 06016 74FC0000 (01360) MOV(OQ), MAAABS(A7,A4)\NOP
A41 06018 00000000 (01361) MOV(R,BULL)
A42 0601A 00C00000 (01362) MON(P,OQ)\NOP
A43 0601C 20092009 (01363) SETTC(AP1)
A44 0601E 00050005 (01364) MOV(P,A5)
A45 06020 00A000A0 (01365) MON(P,M5)
A46 06022 20402040 (01366) SET(AF0)
A47 06024 91100040 (01367) JUMPS(WMLFD,P1)
A48 06026 901E0004 (01368) *
A48 06026 901E0004 (01369) *
A49 06028 001C0000 (01370) *
A49 06028 001C0000 (01371) *
A4A 0602A 001C0000 (01372) * WMLF-APU ABORT ROUTINE
A4B 0602C 000C0000 (01373) * P REGISTER=E(M)
A4C 0602E 901C0049 (01374) *
A4C 0602E 901C0049 (01375) *
A4D 06030 20322032 (01376) #4
A4D 06030 20322032 (01377) MOV(ZERO,OQ)\NOP
A4E 06032 10000053 (01378) MOV(ZERO,OQ)\NOP
A4E 06032 10000053 (01379) MOV(P,OQ)\NOP
A4F 06034 9016004F (01380) JUMPC(04,EO)
A4F 06034 9016004F (01381) WMLFD CLEAR(R1)
A50 06036 10000017 (01382) JUMP(RQUAN)
A51 06038 00000000 (01383) *
A51 06038 00000000 (01384) *
A52 0603A 10000039 (01385) WMLFRM JUMPC(WMLFRM,P1)
A52 0603A 10000039 (01386) JUMP(WMLFRE)
A52 0603A 10000039 (01387) WMLFAV MOV(IGA,MULL)\NOP
A52 0603A 10000039 (01388) JUMP(WMLFAE)
A52 0603A 10000039 (01389) JEND OF WMLD APU PROGRAM
A52 0603A 10000039 (01390) *
A52 0603A 10000039 (01391) *

```

A(M+J)=B
 P(M+J)=B
 E(M+J)=E(M)

JAPU HALTS HERE AT END OF WMLD,
 ; THEN GOES TO EQUAN WHEN RESTARTED
 ; BY THE APS PROGRAM

LET APS/OTPT GO
 GO BACK UNLESS DONE

PAGE 33: (R0WD1)<DC116>B0N1CH.MSD.2, 29-Dec-88 14130140, ED: WOLF
MULQC(V,A,W,V,W) WEINER-LEVINSON MATRIX SOL-W & P-6 CODING/QUANT.

```

(01392) )
(01393) )QUANT: QUANTIZE AND CODE 6 REFLECTION COEFFICIENTS
(01394) ) J. WOLF 7/9/88
(01395) )
(01396) )INPUT AND OUTPUT STREAMS:
(01397) )IQR: 2**15,
(01398) ) FOR EACH COEFF: K(1), (CTH(K),DVL(K)), N=1,...,LENGTH(IW1)
(01399) ) APS SETS AFB ON LAST K(1) (EO WILL BE TWO LATER).
(01400) )QOI: (FLY PT VALUE(1), FID PT CODE(1), I=1,...,6)
(01401) )
(01402) )TABLE LENGTHS FOR K(1) ARE 64, 32, 16, 16, 16, 16
(01403) )APU REGISTER USAGE: CODE CTR
(01404) ) A0: K(1) 2**15
(01405) ) A1: CTH(K)
(01406) ) A2: DVL(K)
(01407) )
(01408) )QUANT: NOP \ MOV(IQR,A1)
(01409) )LOOP FOR EACH NEW COEFFICIENT TO BE QUANTIZED/CODED
(01410) )I: MOV(IQR,A0) \ MOV(ZERO,A0)
(01411) )NOP \ R(A0)
(01412) )INNER LOOP
(01413) )I2: MOV(IQR,A1) \ NOP
(01414) )SUB(AB,A1) \ MOV(AB,A0),ADD(AB,A1)
(01415) )MOV(IQR,A2) \ NOP
(01416) )MOV(R,NOEL) \ NOP
(01417) )JUMPS(RQSDONE,FWI)
(01418) )JUMPC(R2,I1)
(01419) )FOUND THE RIGHT QUANTIZATION BIN. NOW READ IN THE REST OF TABLE
(01420) )I3: MOV(IQR,NOEL) \ NOP
(01421) )NOP
(01422) )JUMPC(R3,FWI)
(01423) )RQSDONE: CLEAR(FWI)
(01424) )R(I2) \ ALIGN(AB)
(01425) )MOV(R,OQ) \ MOV(R,OQ)
(01426) )JUMPC(I1,AFB)
(01427) )CLEAR(RA)
(01428) )JUMP(0)
(01429) )
(01430) )MMLF$SZ=8A-MWLP$SA
(01431) )
(01432) )
(01433) )
(01434) )
(01435) )
(01436) )
(01437) )
(01438) )
(01439) )
(01440) )
(01441) )
(01442) )
(01443) )
(01444) )
(01445) )
(01446) )
(01447) )
(01448) )
(01449) )
(01450) )
(01451) )
(01452) )
(01453) )
(01454) )
(01455) )
(01456) )
(01457) )
(01458) )
(01459) )
(01460) )
(01461) )
(01462) )
(01463) )
(01464) )
(01465) )
(01466) )
(01467) )
(01468) )
(01469) )
(01470) )
(01471) )
(01472) )
(01473) )
(01474) )
(01475) )
(01476) )
(01477) )
(01478) )
(01479) )
(01480) )
(01481) )
(01482) )
(01483) )
(01484) )
(01485) )
(01486) )
(01487) )
(01488) )
(01489) )
(01490) )
(01491) )
(01492) )
(01493) )
(01494) )
(01495) )
(01496) )
(01497) )
(01498) )
(01499) )
(01500) )
(01501) )
(01502) )
(01503) )
(01504) )
(01505) )
(01506) )
(01507) )
(01508) )
(01509) )
(01510) )
(01511) )
(01512) )
(01513) )
(01514) )
(01515) )
(01516) )
(01517) )
(01518) )
(01519) )
(01520) )
(01521) )
(01522) )
(01523) )
(01524) )
(01525) )
(01526) )
(01527) )
(01528) )
(01529) )
(01530) )
(01531) )
(01532) )
(01533) )
(01534) )
(01535) )
(01536) )
(01537) )
(01538) )
(01539) )
(01540) )
(01541) )
(01542) )
(01543) )
(01544) )
(01545) )
(01546) )
(01547) )
(01548) )
(01549) )
(01550) )
(01551) )
(01552) )
(01553) )
(01554) )
(01555) )
(01556) )
(01557) )
(01558) )
(01559) )
(01560) )
(01561) )
(01562) )
(01563) )
(01564) )
(01565) )
(01566) )
(01567) )
(01568) )
(01569) )
(01570) )
(01571) )
(01572) )
(01573) )
(01574) )
(01575) )
(01576) )
(01577) )
(01578) )
(01579) )
(01580) )
(01581) )
(01582) )
(01583) )
(01584) )
(01585) )
(01586) )
(01587) )
(01588) )
(01589) )
(01590) )
(01591) )
(01592) )
(01593) )
(01594) )
(01595) )
(01596) )
(01597) )
(01598) )
(01599) )
(01600) )
(01601) )
(01602) )
(01603) )
(01604) )
(01605) )
(01606) )
(01607) )
(01608) )
(01609) )
(01610) )
(01611) )
(01612) )
(01613) )
(01614) )
(01615) )
(01616) )
(01617) )
(01618) )
(01619) )
(01620) )
(01621) )
(01622) )
(01623) )
(01624) )
(01625) )
(01626) )
(01627) )
(01628) )
(01629) )
(01630) )
(01631) )
(01632) )
(01633) )
(01634) )
(01635) )
(01636) )
(01637) )
(01638) )
(01639) )
(01640) )
(01641) )
(01642) )
(01643) )
(01644) )
(01645) )
(01646) )
(01647) )
(01648) )
(01649) )
(01650) )
(01651) )
(01652) )
(01653) )
(01654) )
(01655) )
(01656) )
(01657) )
(01658) )
(01659) )
(01660) )
(01661) )
(01662) )
(01663) )
(01664) )
(01665) )
(01666) )
(01667) )
(01668) )
(01669) )
(01670) )
(01671) )
(01672) )
(01673) )
(01674) )
(01675) )
(01676) )
(01677) )
(01678) )
(01679) )
(01680) )
(01681) )
(01682) )
(01683) )
(01684) )
(01685) )
(01686) )
(01687) )
(01688) )
(01689) )
(01690) )
(01691) )
(01692) )
(01693) )
(01694) )
(01695) )
(01696) )
(01697) )
(01698) )
(01699) )
(01700) )
(01701) )
(01702) )
(01703) )
(01704) )
(01705) )
(01706) )
(01707) )
(01708) )
(01709) )
(01710) )
(01711) )
(01712) )
(01713) )
(01714) )
(01715) )
(01716) )
(01717) )
(01718) )
(01719) )
(01720) )
(01721) )
(01722) )
(01723) )
(01724) )
(01725) )
(01726) )
(01727) )
(01728) )
(01729) )
(01730) )
(01731) )
(01732) )
(01733) )
(01734) )
(01735) )
(01736) )
(01737) )
(01738) )
(01739) )
(01740) )
(01741) )
(01742) )
(01743) )
(01744) )
(01745) )
(01746) )
(01747) )
(01748) )
(01749) )
(01750) )
(01751) )
(01752) )
(01753) )
(01754) )
(01755) )
(01756) )
(01757) )
(01758) )
(01759) )
(01760) )
(01761) )
(01762) )
(01763) )
(01764) )
(01765) )
(01766) )
(01767) )
(01768) )
(01769) )
(01770) )
(01771) )
(01772) )
(01773) )
(01774) )
(01775) )
(01776) )
(01777) )
(01778) )
(01779) )
(01780) )
(01781) )
(01782) )
(01783) )
(01784) )
(01785) )
(01786) )
(01787) )
(01788) )
(01789) )
(01790) )
(01791) )
(01792) )
(01793) )
(01794) )
(01795) )
(01796) )
(01797) )
(01798) )
(01799) )
(01800) )
(01801) )
(01802) )
(01803) )
(01804) )
(01805) )
(01806) )
(01807) )
(01808) )
(01809) )
(01810) )
(01811) )
(01812) )
(01813) )
(01814) )
(01815) )
(01816) )
(01817) )
(01818) )
(01819) )
(01820) )
(01821) )
(01822) )
(01823) )
(01824) )
(01825) )
(01826) )
(01827) )
(01828) )
(01829) )
(01830) )
(01831) )
(01832) )
(01833) )
(01834) )
(01835) )
(01836) )
(01837) )
(01838) )
(01839) )
(01840) )
(01841) )
(01842) )
(01843) )
(01844) )
(01845) )
(01846) )
(01847) )
(01848) )
(01849) )
(01850) )
(01851) )
(01852) )
(01853) )
(01854) )
(01855) )
(01856) )
(01857) )
(01858) )
(01859) )
(01860) )
(01861) )
(01862) )
(01863) )
(01864) )
(01865) )
(01866) )
(01867) )
(01868) )
(01869) )
(01870) )
(01871) )
(01872) )
(01873) )
(01874) )
(01875) )
(01876) )
(01877) )
(01878) )
(01879) )
(01880) )
(01881) )
(01882) )
(01883) )
(01884) )
(01885) )
(01886) )
(01887) )
(01888) )
(01889) )
(01890) )
(01891) )
(01892) )
(01893) )
(01894) )
(01895) )
(01896) )
(01897) )
(01898) )
(01899) )
(01900) )
(01901) )
(01902) )
(01903) )
(01904) )
(01905) )
(01906) )
(01907) )
(01908) )
(01909) )
(01910) )
(01911) )
(01912) )
(01913) )
(01914) )
(01915) )
(01916) )
(01917) )
(01918) )
(01919) )
(01920) )
(01921) )
(01922) )
(01923) )
(01924) )
(01925) )
(01926) )
(01927) )
(01928) )
(01929) )
(01930) )
(01931) )
(01932) )
(01933) )
(01934) )
(01935) )
(01936) )
(01937) )
(01938) )
(01939) )
(01940) )
(01941) )
(01942) )
(01943) )
(01944) )
(01945) )
(01946) )
(01947) )
(01948) )
(01949) )
(01950) )
(01951) )
(01952) )
(01953) )
(01954) )
(01955) )
(01956) )
(01957) )
(01958) )
(01959) )
(01960) )
(01961) )
(01962) )
(01963) )
(01964) )
(01965) )
(01966) )
(01967) )
(01968) )
(01969) )
(01970) )
(01971) )
(01972) )
(01973) )
(01974) )
(01975) )
(01976) )
(01977) )
(01978) )
(01979) )
(01980) )
(01981) )
(01982) )
(01983) )
(01984) )
(01985) )
(01986) )
(01987) )
(01988) )
(01989) )
(01990) )
(01991) )
(01992) )
(01993) )
(01994) )
(01995) )
(01996) )
(01997) )
(01998) )
(01999) )
(02000) )

```


HWL06-APS PROGRAM

176

```

00000100 (01649) *
00000100 (01650) MULTPSA=BC
00000100 (01651) *
00102 (01652) * END
00102 (01653) *
00102 (01654) *
00102 (01655) *
00102 00000000 (01656) MULTPSI DATA 0P'A.0"
...
00000100 (01657) *
00000100 (01658) MULTPSZ=0L-MULTSAPS

```


PAGE 39: (000007)DCAL000000100.2, 29-Dec-88 14130140, R4: WULF
 MWLQSSM - SPECIAL INTERMEDIATE SUPPORT FOR MWLQ6

06172 06006170	(01659)	• MWLQSSM - SPECIAL INTERMEDIATE SUPPORT FOR MWLQ6	
06174 0670	(01660)	•	
06175 00000001	(01661)	MWLQSSM:	
	(01662)	CALL R6, MWSS1	
	(01663)	RET	
	(01664)	JMP APSONE	
	(01665)	•	
	(01666)	•	
06177 0000	(01667)	EVEN	
06178 06004202	(01668)	MWSS1: MOVNR R6, INST	
0617A 06706203	(01669)	MOVNR R7, INST+06	
0617C 046177C0	(01670)	MOVNR R6, APS0LA	
0617E 066377C0	(01671)	MOVNR R31, SVSSPLGS	
06200 0070	(01672)	RETURN	
	(01673)	•	
06201 0000	(01674)	EVEN	
	(01675)	BEGIN APS(F00)	
06202 00000020	(01676)	HOP(0)	
06204	(01677)	P00	

JLOAD HOP(0) INTO APS LOC 0
 JSET R1
 JTHIS INSTR GETS PUT AT 0A=0

```

(01678) * MEWL(V,I,U,V,W)
(01679) * CORRELATION MATRIX EQUATION, WEINER-LEVINSON SOLUTION
(01680) *
(01681) * SEPTEMBER 14, 1979
(01682) *
(01683) * BUFFER USAGE
(01684) *
(01685) * V(K), OUTPUT VECTOR
(01686) * W(K), INPUT VECTOR
(01687) * V(K)=R(K)-R(K), CORRELATION MATRIX ELEMENTS
(01688) * W(K), WORKING BUFFER
(01689) * (I), (I-1), INTEGER TABLE OUTPUTS
(01690) * (N); THRESHOLD, EARLY TERMINATION TEST.
(01691) *
(01692) * FUNCTION W(K)=SUM J=C0,N-13 OF R(K-J)*V(J)
(01693) * (I)=Y2-1-N,
(01694) * (I+1)=E(N)/R(N), RELATIVE ENERGY, "SPIKE SOLUTION"
(01695) * IF COMPLETED SUCCESSFULLY, N=Y2-1, I.E. (I)=0
(01696) *
(01697) * ALGORITHM
(01698) * DETERMINES SOLUTION BY FORWARD INTERATION
(01699) * MAKES USE OF THE SPIKE CASE, I.E.
(01700) * -R(K+1)=SUM FOR J=C0,N-13 OF R(K-J)*A(N-1,J)
(01701) *
(01702) * INITIALIZATION: E(-1)=R(N)
(01703) * RECURSION FORMULA
(01704) *
(01705) * B(N)=R(N+1)+SUM J=C0,N-13 OF R(N-J)*A(N-1,J)
(01706) * C(N)=U(N)-SUM J=C0,N-13 OF R(N-J)*V(N-1,J)
(01707) *
(01708) * KB(N), REFLECTANCE, =R(N)/E(N-1)
(01709) * KC(N)=C(N)/E(N-1)
(01710) * E(N), ENERGY, SPIKE CASE, = E(N-1)-B(N)*KB(N)
(01711) *
(01712) * V(N,J)=V(N-1,J)+EC(N)*A(N-1,N-1-J) V(N,N)=EC(N)
(01713) * A(N,J)=A(N-1,J)-KB(N)*A(N-1,N-1-J) A(N,N)=E(N)
(01714) *
(01715) * THE EXTERNAL VALUES
(01716) * A(N,-1)=1, A(N,J)=V(N,J)=0, J>N
(01717) * ARE CONSISTENT WITH THE RECURSION FORMULA
(01718) *
(01719) * THE ITERATION NORMALLY TERMINATES WITH N=Y2-1
(01720) * EARLY TERMINATION OCCURS IF (0)*MAC (R(N)) > E(N-1)
(01721) * IN SUCH CASE UNUSED VALUES OF A,K0, AND Y ARE SET TO ZERO
(01722) *
(01723) * BUFFER USAGE
(01724) *
(01725) * V(K)=V(N,K)
(01726) * W(K)=A(N,K), 0<=K<N
(01727) * W(N+E)=KB(E), 0<=E<N
(01728) * NORMAL TERMINATION, N=Y2-1
(01729) * EARLY TERMINATION, UNUSED VALUES ZERO FILLED
(01730) * (I)=Y2-1-N, N=Y2-1, NORMAL TERMINATION
(01731) * (I+1)=E(N)/R(N)=E(N)/E(-1), FRACTIONAL ENERGY

```

PAGE 41: (0000) <OCALIG>ENH16M.H50.2, 29-Dec-88 14:38:48, Ed: WOLF
NEWL(Y,I,W,D,V,U)

PAGE 43: (BMM030C0A16)B0016M.MS0.2, 29-Dec-88 14130140, Ed: WOLF
MEWL - APB PROGRAM

```

A17 06234 04000000 (01704) MOV(A0),MWL(M1,M5) \ NOP JAB=A(M-1,J)*E(M-J)
      (01705) JUMPC(B2,FV1) LOOP UNLESS DONE
A18 06236 90160010 (01706) CLRZ(A0)
A19 06238 20372037 (01707) MEWL3: MOV(A4),ADD(A0,A5) \ NOP J44=CSUM
      (01708) MOV(P,A1) \ NOP J41=Y(M-1,N-1)*R(1)
A1A 0623A 45140000 (01709) MOV(A5),SUB(A4,A1) \ NOP J45=B(M)
A1B 0623C 00010000 (01790) MOV(A4),R(A6) \ NOP J44=C(M), SET REGC=E(M-1)
A1C 0623E 49950000 (01791)
A1D 06240 02040000 (01792)
      (01793)
      (01794)
      (01795)
      (01796)
      (01797) REFLECTANCE CALCULATION AND EARLY TERMINATION TEST
      (01798) FUNCTION
      (01799) IF (0)*ABS B(M) > E(M-1), EARLY TERMINATION, SET AP2
      (01800) A7=A7-1, DECREMENT COUNT
      (01801) R=1/R(M-1)
      (01802)
      (01803) RB(M)=R*B(M), KC(M)=R*C(M),E(M)=E(M-1)-0(M)*RB(M)
      (01804)
      (01805) SEQUENCING
      (01806) INPUT (0),INTEGER
      (01807) OUT RB(M), OUT A(M,N)--RB(N),OUT V(M,N)=EC(M)
      (01808)
      (01809) REGISTER CONTENTS
      (01810) A4=C(M), A5=0(M), A6=REGC=E(M-1),A7=YZ-M
      (01811)
      (01812) PART1: RECIPROCAL AND TERMINATION TEST
      (01813)
      (01814) MEWL4: MOV(M0),RCP(A6) \ NOP M0=E(M-1)
      (01815) MOV(M4),K(2) \ NOP M4=RB=RCPE(M-1)
      (01816) MUL(M0,M4) \ NOP
      (01817)
      (01818) MOV(A1),R(A5) \ NOP J41=2
      (01819) MOV(M2),R(A4) \ NOP J42=B(M)
      (01820) MOV(P,A0) \ NOP J40=RB*RCPE(M-1)
      (01821) MOV(M3),SUB(A1,A0) \ NOP J43=C(M)
      (01822) MOV(M,M1) \ NOP J41=2-M0*RCPE(M-1)
      (01823)
      (01824) MUL(M1,M4) \ NOP
      (01825) MOV(TQA,M7) \ NOP J47=(0), THRESHOLD
      (01826) MOV(P,M4) \ NOP J40=R1=RB*(2-R0)*E(M-1)
      (01827) MUL(M0,M4) \ NOP
      (01828)
      (01829) MOV(TQA,A3) \ NOP J43=INTEGER 1
      (01830) SUR(A7,A3) \ NOP J47=YZ-M-1
      (01831) MOV(A0),ABS MUL(M2,M7) \ NOP J40=R1=E(M-1)
      (01832)
      (01833) MOV(A7),SUB(A1,A0) \ NOP J47=YZ-M-1
      (01834) MOV(R,M1) \ NOP J41=2-R1+E(M-1)
      (01835) MOV(A0),MUL(M1,M4) \ NOP J40=(0)*B(M)
      (01836)

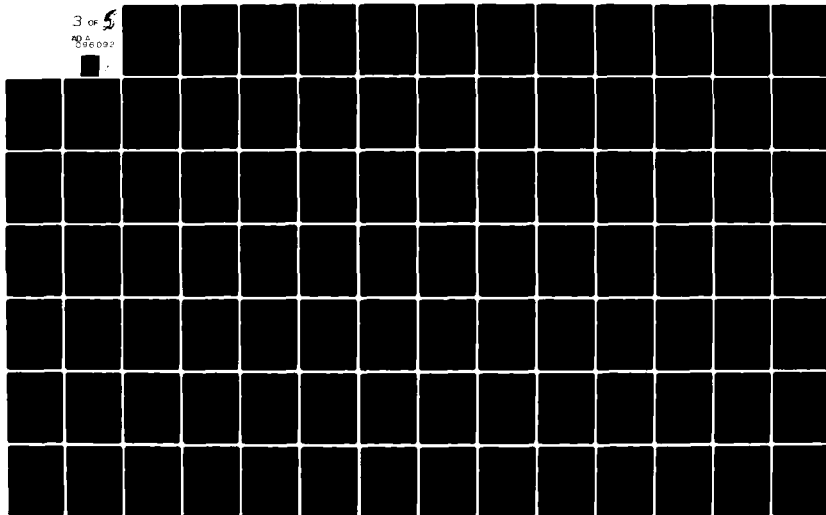
```


AD-A096 092

BOLT BERANEK AND NEWMAN INC CAMBRIDGE MA
DESIGN AND REAL-TIME IMPLEMENTATION OF A ROBUST APC CODER FOR S--ETC(U)
DEC 80 J J WOLF, K D FIELD, W H RUSSELL
DCA100-79-C-0037
NL

UNCLASSIFIED

3 OF 5
NO 5
096092



```

(01898) ) A(M,N)=-KB(M) ) Y(M,N)=KC(M),
(01891) ) WERE EXECUTED DURING REFLECTANCE CALCULATION
(01892) ) A(M,J)=A(M-1,J)-KB(M)*A(M-1,M-1-J)
(01893) ) A(M,M-1-J)=A(M-1,M-1-J)-KB(M)*A(M-1,J)
(01894) ) Y(M,J)=Y(M-1,J)+KC(M)*A(M-1,M-1-J)
(01895) ) Y(M,M-1-J)=Y(M,M-1-J)+KC(M)*A(M-1,J)
(01896) ) AS INDICATED, RECURSION WORKS WITH ENDS TOWARDS MIDDLE
(01897) ) FOR N ODD, RECURSION IS DUPLICATED ON J=(N+1)/2 VALUE
(01898) )
SEQUENCING
(01899) ) A(M-1,M-1),A(M-1,M), JUMP TO FIRST Y INPUT IN LOOP
(01900) ) )
(01901) ) )
(01902) ) )
(01903) ) )
(01904) ) )
(01905) ) )
(01906) ) )
(01907) ) )
(01908) ) )
(01909) ) )
(01910) ) )
(01911) ) )
(01912) ) )
(01913) ) )
(01914) ) )
(01915) ) )
(01916) ) )
(01917) ) )
(01918) ) )
(01919) ) )
(01920) ) )
(01921) ) )
(01922) ) )
(01923) ) )
(01924) ) )
(01925) ) )
(01926) ) )
(01927) ) )
(01928) ) )
(01929) ) )
(01930) ) )
(01931) ) )
(01932) ) )
(01933) ) )
(01934) ) )
(01935) ) )
(01936) ) )
(01937) ) )
(01938) ) )
(01939) ) )
(01940) ) )
(01941) ) )
(01942) ) )
(01943) ) )
(01944) ) )
(01945) ) )
(01946) ) )
(01947) ) )
(01948) ) )
(01949) ) )
(01950) ) )
(01951) ) )
(01952) ) )
(01953) ) )
(01954) ) )
(01955) ) )
(01956) ) )
(01957) ) )
(01958) ) )
(01959) ) )
(01960) ) )
(01961) ) )
(01962) ) )
(01963) ) )
(01964) ) )
(01965) ) )
(01966) ) )
(01967) ) )
(01968) ) )
(01969) ) )
(01970) ) )
(01971) ) )
(01972) ) )
(01973) ) )
(01974) ) )
(01975) ) )
(01976) ) )
(01977) ) )
(01978) ) )
(01979) ) )
(01980) ) )
(01981) ) )
(01982) ) )
(01983) ) )
(01984) ) )
(01985) ) )
(01986) ) )
(01987) ) )
(01988) ) )
(01989) ) )
(01990) ) )
(01991) ) )
(01992) ) )
(01993) ) )
(01994) ) )
(01995) ) )
(01996) ) )
(01997) ) )
(01998) ) )
(01999) ) )
(02000) ) )
(02001) ) )
(02002) ) )
(02003) ) )
(02004) ) )
(02005) ) )
(02006) ) )
(02007) ) )
(02008) ) )
(02009) ) )
(02010) ) )
(02011) ) )
(02012) ) )
(02013) ) )
(02014) ) )
(02015) ) )
(02016) ) )
(02017) ) )
(02018) ) )
(02019) ) )
(02020) ) )
(02021) ) )
(02022) ) )
(02023) ) )
(02024) ) )
(02025) ) )
(02026) ) )
(02027) ) )
(02028) ) )
(02029) ) )
(02030) ) )
(02031) ) )
(02032) ) )
(02033) ) )
(02034) ) )
(02035) ) )
(02036) ) )
(02037) ) )
(02038) ) )
(02039) ) )
(02040) ) )
(02041) ) )
(02042) ) )
(02043) ) )
(02044) ) )
(02045) ) )
(02046) ) )
(02047) ) )
(02048) ) )
(02049) ) )
(02050) ) )
(02051) ) )
(02052) ) )
(02053) ) )
(02054) ) )
(02055) ) )
(02056) ) )
(02057) ) )
(02058) ) )
(02059) ) )
(02060) ) )
(02061) ) )
(02062) ) )
(02063) ) )
(02064) ) )
(02065) ) )
(02066) ) )
(02067) ) )
(02068) ) )
(02069) ) )
(02070) ) )
(02071) ) )
(02072) ) )
(02073) ) )
(02074) ) )
(02075) ) )
(02076) ) )
(02077) ) )
(02078) ) )
(02079) ) )
(02080) ) )
(02081) ) )
(02082) ) )
(02083) ) )
(02084) ) )
(02085) ) )
(02086) ) )
(02087) ) )
(02088) ) )
(02089) ) )
(02090) ) )
(02091) ) )
(02092) ) )
(02093) ) )
(02094) ) )
(02095) ) )
(02096) ) )
(02097) ) )
(02098) ) )
(02099) ) )
(02100) ) )
(02101) ) )
(02102) ) )
(02103) ) )
(02104) ) )
(02105) ) )
(02106) ) )
(02107) ) )
(02108) ) )
(02109) ) )
(02110) ) )
(02111) ) )
(02112) ) )
(02113) ) )
(02114) ) )
(02115) ) )
(02116) ) )
(02117) ) )
(02118) ) )
(02119) ) )
(02120) ) )
(02121) ) )
(02122) ) )
(02123) ) )
(02124) ) )
(02125) ) )
(02126) ) )
(02127) ) )
(02128) ) )
(02129) ) )
(02130) ) )
(02131) ) )
(02132) ) )
(02133) ) )
(02134) ) )
(02135) ) )
(02136) ) )
(02137) ) )
(02138) ) )
(02139) ) )
(02140) ) )
(02141) ) )
(02142) ) )
(02143) ) )
(02144) ) )
(02145) ) )
(02146) ) )
(02147) ) )
(02148) ) )
(02149) ) )
(02150) ) )
(02151) ) )
(02152) ) )
(02153) ) )
(02154) ) )
(02155) ) )
(02156) ) )
(02157) ) )
(02158) ) )
(02159) ) )
(02160) ) )
(02161) ) )
(02162) ) )
(02163) ) )
(02164) ) )
(02165) ) )
(02166) ) )
(02167) ) )
(02168) ) )
(02169) ) )
(02170) ) )
(02171) ) )
(02172) ) )
(02173) ) )
(02174) ) )
(02175) ) )
(02176) ) )
(02177) ) )
(02178) ) )
(02179) ) )
(02180) ) )
(02181) ) )
(02182) ) )
(02183) ) )
(02184) ) )
(02185) ) )
(02186) ) )
(02187) ) )
(02188) ) )
(02189) ) )
(02190) ) )
(02191) ) )
(02192) ) )
(02193) ) )
(02194) ) )
(02195) ) )
(02196) ) )
(02197) ) )
(02198) ) )
(02199) ) )
(02200) ) )
(02201) ) )
(02202) ) )
(02203) ) )
(02204) ) )
(02205) ) )
(02206) ) )
(02207) ) )
(02208) ) )
(02209) ) )
(02210) ) )
(02211) ) )
(02212) ) )
(02213) ) )
(02214) ) )
(02215) ) )
(02216) ) )
(02217) ) )
(02218) ) )
(02219) ) )
(02220) ) )
(02221) ) )
(02222) ) )
(02223) ) )
(02224) ) )
(02225) ) )
(02226) ) )
(02227) ) )
(02228) ) )
(02229) ) )
(02230) ) )
(02231) ) )
(02232) ) )
(02233) ) )
(02234) ) )
(02235) ) )
(02236) ) )
(02237) ) )
(02238) ) )
(02239) ) )
(02240) ) )
(02241) ) )
(02242) ) )
(02243) ) )
(02244) ) )
(02245) ) )
(02246) ) )
(02247) ) )
(02248) ) )
(02249) ) )
(02250) ) )
(02251) ) )
(02252) ) )
(02253) ) )
(02254) ) )
(02255) ) )
(02256) ) )
(02257) ) )
(02258) ) )
(02259) ) )
(02260) ) )
(02261) ) )
(02262) ) )
(02263) ) )
(02264) ) )
(02265) ) )
(02266) ) )
(02267) ) )
(02268) ) )
(02269) ) )
(02270) ) )
(02271) ) )
(02272) ) )
(02273) ) )
(02274) ) )
(02275) ) )
(02276) ) )
(02277) ) )
(02278) ) )
(02279) ) )
(02280) ) )
(02281) )
```


PAGE 46: [RMD]<DCAL6>RBN16M.MSD.2, 29-Dec-88 14:38:48, ED: WOLF
MEVL - LPU PROGRAM

```

A66 062D2 447C0000 (01943)      MOV(QQ),ADD(A3,A4) \ NOP
A67 062D4 089C0000 (01944)      MOV(R,QQ) \ NOP
A68 062D6 90A00066 (01946) MEVL7: JUMPC(R1,AF2)
                                (01947) \ TERMINATION
                                (01948) \ FUNCTION
                                (01949) \ (1)=YZ-I-M, INTEGER
                                (01950) \ (I+1)=E(M-1)/R(M), FRACTION, FIXED
                                (01951) \ V,A,K0,ZERO FILLED IF NECESSARY
                                (01952) \ SEQUENCING
                                (01953) \ R(0),INTEGER1,OUT(I), OUT(I+1),
                                (01954) \ OUT V,A,K0,AS NECESSARY
                                (01955) \ REGISTER CONTENTS
                                (01956) \ A6=E(N-1), A7=VZ-2-M
                                (01957) \
                                (01958) \
                                (01959) \
                                (01960) \
                                (01961) \
                                (01962) \
                                (01963) \
                                (01964) \
                                (01965) \
                                (01966) \
                                (01967) \
                                (01968) \
                                (01969) \
                                (01970) \
                                (01971) \
                                (01972) \
                                (01973) \
                                (01974) \
                                (01975) \
                                (01976) \
                                (01977) \
                                (01978) \
                                (01979) \
                                (01980) \
                                (01981) \
                                (01982) \
                                (01983) \
                                (01984) \
                                (01985) \
                                (01986) MEVL$S2=8A-MEVL$SA
                                (01987) END
                                (01988) \
                                (01989) \

A69 062D8 00000000 (01960)      MOV(IQ,AB) \ NOP
A6A 062DA 2E000000 (01961)      RCP(AB) \ NOP
A6B 062DC 00E00000 (01962)      MOV(IA,M0) \ NOP
A6C 062DE 16AC0000 (01963)      MOV(M4),K(2) \ NOP
A6D 062F0 04000000 (01964)      MUL(M0,M4) \ NOP
                                (01965) \
                                (01966) \
                                (01967) \
                                (01968) \
                                (01969) \
                                (01970) \
                                (01971) \
                                (01972) \
                                (01973) \
                                (01974) \
                                (01975) \
                                (01976) \
                                (01977) \
                                (01978) \
                                (01979) \
                                (01980) \
                                (01981) \
                                (01982) \
                                (01983) \
                                (01984) \
                                (01985) \
                                (01986) MEVL$S2=8A-MEVL$SA
                                (01987) END
                                (01988) \
                                (01989) \

A6E 062E2 00070000 (01966)      MOV(P,A2) \ NOP
A6F 062F4 4A310000 (01967)      MOV(A1),SUB(A1,A2) \ NOP
A70 062E6 3AE90000 (01968)      MOV(M1),ALIGN(A7) \ NOP
A71 062F8 04000000 (01969)      MUL(M1,M4) \ NOP
                                (01970) \
                                (01971) \
                                (01972) \
                                (01973) \
                                (01974) \
                                (01975) \
                                (01976) \
                                (01977) \
                                (01978) \
                                (01979) \
                                (01980) \
                                (01981) \
                                (01982) \
                                (01983) \
                                (01984) \
                                (01985) \
                                (01986) MEVL$S2=8A-MEVL$SA
                                (01987) END
                                (01988) \
                                (01989) \

A72 062EA 02DC0000 (01971)      MOV(O0),R(A6) \ NOP
A73 062EC 00000000 (01972)      MOV(R,M5) \ NOP
A74 062EE 00A90000 (01973)      MOV(P,M1) \ NOP
A75 062F0 04A00000 (01974)      MUL(M1,M5) \ NOP
                                (01975) \
                                (01976) \
                                (01977) \
                                (01978) \
                                (01979) \
                                (01980) \
                                (01981) \
                                (01982) \
                                (01983) \
                                (01984) \
                                (01985) \
                                (01986) MEVL$S2=8A-MEVL$SA
                                (01987) END
                                (01988) \
                                (01989) \

A76 062F2 00000000 (01976)      MOV(P,A0) \ NOP
A77 062F4 3A000000 (01977)      ALIGN(A0) \ NOP
A78 062F6 009C0000 (01978)      MOV(R,QQ) \ NOP
                                (01979) \
                                (01980) \
                                (01981) \
                                (01982) \
                                (01983) \
                                (01984) \
                                (01985) \
                                (01986) MEVL$S2=8A-MEVL$SA
                                (01987) END
                                (01988) \
                                (01989) \

A79 062F8 001C0000 (01980)      MOV(ZERO,00) \ NOP
A7A 062FA 9A100079 (01981)      JUMPC(RA-1,F1)
A7B 062FC 20320032 (01982)      CLEAR(RA)
A7C 062FE 00000000 (01983)      NOP
A7D 06300 10000000 (01984)      JUMP(MEVL$SA)
                                (01985) \
                                (01986) MEVL$S2=8A-MEVL$SA
                                (01987) END
                                (01988) \
                                (01989) \

06302 0000007E (01986) MEVL$S2=8A-MEVL$SA
                                (01987) END
                                (01988) \
                                (01989) \

```


188

PAGE 581 (0000)C0C16>00016M.MSO.2, 29-Dec-88 14:30:40, Ed: WOLF
NEWL - APS PROGRAM

```

(02135) ,
A3P 06394 7F410013      MOVW(BW0,BR1,TP)      ;BW0=A(N-N/2), OUTPUT
A40 06396 80010015      MOVW(BW0,BR2,TP)      ;BW0=Y(N/2-1), OUTPUT
A41 06398 02010017      MOVW(BW0,BR3,TP)      ;BW0=Y(N-N/2), OUTPUT
A42 0639A 04000A60      JUMP(NEWL15)
(02141) ,
(02142) ,
(02143) ,
(02144) ,
(02145) ,
(02146) ,
(02147) ,
(02148) ,
(02149) ,
(02150) ,
(02151) ,
(02152) ,
(02153) ,
(02154) ,
(02155) ,
(02156) ,
(02157) ,
(02158) ,
(02159) ,
(02160) ,
(02161) ,
(02162) ,
(02163) ,
(02164) ,
(02165) ,
(02166) ,
(02167) ,
(02168) ,
(02169) ,
(02170) ,
(02171) ,
(02172) ,
(02173) ,
(02174) ,
(02175) ,
(02176) ,
(02177) ,
(02178) ,
(02179) ,
A43 0639C 06910000      NEWL16:00(BR1,N,TP)      ;BR0=R(0), INPUT
A44 0639E 0852630A      LOAD(BR1,IT(1),L)      ;BR1=1 ADDRESS
A45 063A0 00010013      MOVW(BW0,BR1,TP)      ;OUTPUT (1)=YZ-1-N
A46 063A2 0C70001C      NEWL17:LOAD(BR3,2*YZ)      ;BR3=2*YZ
A47 063A4 0E390026      SUBW(BR3,BW3)          ;BR3=2*(YZ-N-1)
(02150) ,
(02151) ,
(02152) ,
(02153) ,
(02154) ,
(02155) ,
(02156) ,
(02157) ,
(02158) ,
(02159) ,
(02160) ,
(02161) ,
(02162) ,
(02163) ,
(02164) ,
(02165) ,
(02166) ,
(02167) ,
(02168) ,
(02169) ,
(02170) ,
(02171) ,
(02172) ,
(02173) ,
(02174) ,
(02175) ,
(02176) ,
(02177) ,
(02178) ,
(02179) ,
A48 063A6 9009002E      ADDW(BR0,BW3)          ;BR0=A(N-1)
A49 063A8 9229002E      ADDW(BR2,BW3)          ;BR2=Y(N-1)
A4A 063AA 94110011      MOVW(BW1,BR0)          ;BW1=A(N-1)
A4B 063AC 96310015      MOVW(BW3,BR2)          ;BW3=Y(N-1)
A4C 063AE 99015079      ADDL(BW0,1,TP),JUMP(NEWL19) ;OUT (1+1)=E(N)/R(N)
A4D 063B0 9A91003A      ADDL(BW1,2,TP)          ;ZERO FILL A(N-1+J)
A4E 063B2 9CA1003A      ADDL(BW2,2,TP)          ;ZERO FILL R0(N-1+J)
A4F 063B4 9EB1003A      ADDL(BW3,2,TP)          ;ZERO FILL Y(N-1+J)
A50 063B6 A0394002      NEWL19:SUBL(BR3,2),JUMPP(R10)
A51 063B8 A2200031      CLEAR(R1)
A52 063BA A4000020      NOP(0)
(02174) ,
(02175) ,
(02176) ,
(02177) ,
(02178) ,
(02179) ,
063BC 00000000      NEWLSC=BC
063BC 000000A6      END
063BC 000000A6      NEWL$Z=BL-NEWL$
063BC 000000BC      NEWL11=BL
063BC 0001      DATA $0001

```

```

(02180) * SBMSMEVL - SPECIAL SUPPORT ROUTINE FOR MEVL
(02181) ;
(02182) ;
063BD C63408E8 (02183) SBMSMEVL PUSHMIL R3,AFDT$ORC(R2)      STACK APU MODULE BUS ORIGIN
MOVIR R7,-4$
063BE 9871FFFE (02184) PUSHMIL R3,0-W$(R3)      STACK APU MODULE ST ADDR AND SZ
063C1 C607FFFE (02185) PUSHMIL R3,AFDT$ORC+W$(R2)  STACK APS MODULE BUS ORIGIN
063C3 C63408E8 (02186) PUSHMIL R3,AFDT$ORC+2*W$(R2)  STACK CSPU SUPPORT MODULE
063C5 C63408E8 (02187) DECR R7,1
063C7 2771 (02188) EVEN
063C8 C407FFFC (02189) PUSHMIL R3,0-2*W$(R3)  APS MODULE SIZE
063CA 9871FFFD (02190) MOVIR R4,-3*W$(R3)  GET SPECIAL SUPPORT SEMAPHORE
063CC 7840FF88 (02191) MOVIR R4,R4,MSK$LOVT
063CE 3C48 (02192) LRS R4,8
063CF 3438 (02193) PUSHMIL R3,R4
063D0 4052 (02194) MOVIR R5,R1
063D1 2651 (02195) INCR R5,W$
063D2 784100FF (02196) MOVIR R4,R5,MSK$RBYT
063D3 3638 (02197) PUSHMIL R3,R4
063D5 C6380792 (02198) PUSHMIL R3,CLR$C0G1
063D7 2632 (02199) INCR R3,2
063D8 C638078A (02200) PUSHMIL R3,ZERO
063DA 9F380811 (02201) SUBIR R3,AP$LL0C
(02202) ;
(02203) ;
(02204) ;
063DC F8420802 (02205) * SCALAR BINDING
MOVIR R4,2*W$(R1)
063DE 914008FF (02206) ANDIR R4,MSK$RBYT
063E0 3A41 (02207) LLS R4,1
063E1 9C400302 (02208) ADDIR R4,SVTS
063E3 E8406351 (02209) MOVIR R4,MEVL$+W$-NEWLAI1+W$  STORED APS INSTR IN MAIN MEM
(02210) ;
(02211) ;
063E5 F8420801 (02212) * INTEGER TABLE BINDING (SCALAR A)
MOVIR R4,W$(R1)
063E7 9A4008FF (02213) ANDIR R4,MSK$RBYT
063E9 9C400502 (02214) ADDIR R4,TSVTS
063EB E840639F (02215) MOVIR R4,MEVL$+W$-NEWLAI6+3*W$  STORED APS INSTR IN MAIN MEM
(02216) ;
(02217) ;
063ED F8620801 (02218) * YBASE BINDING
MOVIR R6,W$(R1)
063EF 586CFF08 (02219) MOVIR R6,R6,MSK$LOVT
063F1 3C67 (02220) LRS R6,7
063F2 C84C0502 (02221) EVEN
063F4 F8706338 (02222) MOVIR R4,ACT$BA(R6)
063F6 564FFFF8 (02223) MOVIR R7,MEVL$+W$-NEWLA7
063F8 84406338 (02224) MOVIR R4,R7,SVTS
063F9 84406338 (02225) MOVIR R4,MEVL$+W$-NEWLA7
(02226) ;
(02227) ;
063FA F84C0605 (02228) * VZ BINDING
MOVIR R4,W$+BCT$AD(R6)
063FC 2641 (02229) INCR R4,1
063FD E8406338 (02230) MOVIR R4,MEVL$+W$-NEWLAI+W$
063FF 3A41 (02231) LLS R4,1
(02232) ;

```

191

```

(02284) * APU3 - STAB(Y,A,U,V)
(02285) * WNR 0/00
(02286) *
(02287) * ROUTINE TO CHECK STABILITY OF
(02288) * PITCH PREDICTOR COEFFICIENTS
(02289) * ALSO CODES AND QUANTIZES COEFF'S
(02290) *
(02291) * THE INPUT COEFFICIENTS ARE : V(0),V(1),V(2)
(02292) * THEY ARE LINEARLY TRANSFORMED AS:
(02293) *
(02294) * T1=V(0)+V(1)+V(2)
(02295) * T2=V(0)-2*V(1)+V(2)
(02296) * T3=V(0)-V(2)
(02297) *
(02298) * THEN , IF (T1<0 .OR.
(02299) * T2<0 .OR.
(02300) * T3<0 .OR.
(02301) * T2>0 .OR.
(02302) * T3>0 )
(02303) * THE FILTER IS DECLARED UNSTABLE
(02304) * AND THE ONE-TAP FILTER IS USED AS
(02305) * C1=0.
(02306) * C2=SA , SA=ONE-TAP COEFF
(02307) * C3=0.
(02308) * OTHERWISE THE
(02309) * FILTER IS DECLARED STABLE AND
(02310) * C1=V(0)
(02311) * C2=V(1)
(02312) * C3=V(2)
(02313) *
(02314) * CODE AND QUANTIZE C1,C2,C3
(02315) * SET Y-BUFFER OF QUANTIZED VALUES
(02316) * SET D-BUFFER OF CODED VALUES
(02317) *
(02318) * APU REGISTER USAGE
(02319) * LEFT RIGHT
(02320) * A0 V(0):C1 V(0):C1
(02321) * A1 V(1):C2 V(1):C2
(02322) * A2 V(2):C3 V(2):C3
(02323) * A3 T1 T2
(02324) * A4 T3
(02325) * A5 +1:C1,C2,C3 +1:-2:CODE
(02326) * A6 CTHC*(N) 2**15
(02327) * A7 SA:DVLC*(N) SA:
(02328) *
(02329) * INPUT AND OUTPUT SEQUENCE
(02330) * IQ: V(0),V(1),V(2),SA,2**15
(02331) * ,CTHC*(N),DVLC*(N), N=1,...,LENGTH(W1)
(02332) *
(02333) * UQ: (FLT PT QUANTIZED VALUE(1),FIX PT CODE(1),I=1,2,3)
(02334) *
(02335) *
(02336) * STAB-APU INITIALIZATION

```


PAGE 55: (0000)DCAL01000016M.WS0.2, 29-Dec-88 14:30:48, Ed: WOLF
AP03 - STAR(V,A,U,V)

```

A19 06476 00120012 (02390)      MOV(ZERO,A2)      JAZ=C3=0.0
A1A 06478 00910091 (02391)      MOV(R,A1)        JAI=C2=SA=1-TAP COEFF..
A1B 0647A 00000076 (02392)      *START OF QUANTIZATION / CODING
A1C 0647C 02000000 (02393)      STAB$OT: NOP\MOV(IQA,A6)  JHOP\2=0-15 FOR FIXED PT COUNTING
A1D 0647E 30000024 (02394)      * CALL SUBROUTINE CQ$SB FOR EACH COEFF
A1E 06480 02200000 (02395)      * 1ST TAP INPUT
A1F 06482 30000024 (02396)      R(A0)\NOP      JR=AP=CIL\NOP
A20 06484 02400000 (02397)      CALL(CQ$SB)    JCALL QUANTIZATION SUBROUTINE
A21 06486 30000024 (02398)      * 2ND TAP INPUT
A22 06488 20322032 (02399)      R(A1)\NOP      JR=AI=C2 \ NOP
A23 0648A 10000000 (02400)      CALL(CQ$SB)    JCALL QUANTIZATION SUBROUTINE
A24 0648C 00950015 (02401)      * 3RD TAP INPUT
A25 0648E 00000210 (02402)      R(A2)\NOP      JR=A2=C3 \ NOP
A26 06490 00760000 (02403)      CALL(CQ$SB)    JCALL QUANTIZATION SUBROUTINE
A27 06492 4E046B5 (02404)      CLEAR(RA)      JHALT APV
A28 06494 00770000 (02405)      JUMP(R)        JUMP(R)
A29 06496 00800000 (02406)      * * CQ$SB - QUANTIZATION SUBROUTINE
A2A 06498 00860000 (02407)      * PUT COEFF IN A5 AND INIT CODE
A2B 0649A 901F0026 (02408)      CQ$SB: MOV(R,A5)\MOV(ZERO,A5) JINPUT COEFF \ INIT CODE=0
A2C 0649C 00E000E0 (02409)      * INNER LOOP - FIND QUANT VALUE AND CODE
A2D 0649E 00000000 (02410)      MOV(IQA,A6)\NOP JHOP\R=A5=ZERO
A2E 064A0 9016002C (02411)      SUB(A5,A6)\MOV(A5),ADD(A5,A6) JINPUT BOUNDARY - CTH(N)\NOP
A2F 064A2 20372037 (02412)      MOV(IQA,A7)\NOP J(C-CTH(N))\UPDATE CODE, CODE+1
A30 064A4 02003A00 (02413)      JUMP(R,NULL)\NOP JINPUT QUANTIZED VALUE DVL(N)\NOP
A31 064A6 009C009C (02414)      JUMP(START$OT,FWI) JENSURE T01 SET
A32 064A8 00000000 (02415)      JUMP(C#6,T1)    JHOP IF DONE TABLE
A33 064AA 00000000 (02416)      * FOUND THE RIGHT QUANTIZATION BIN. READ IN REST OF TABLE.
A34 064AC 00E000E0 (02417)      P7: MOV(IQA,NULL) JINPUT REST OF TABLE
A35 064AE 00000000 (02418)      JUMP(C#7,FWI)   JTO ALLOW FWI TO SET
A36 064B0 9016002C (02419)      * TABLE FINISHED - OUTPUT DATA
A37 064B2 20372037 (02420)      STAB$OT: CLEAR(WI) JLET APS INPUT PRG PROCEED
A38 064B4 02003A00 (02421)      MOV(R,DQ)\MOV(R,DQ) JQUANT VALUE\FIX CODE VALUE
A39 064B6 009C009C (02422)      RETURN          JOUTPUT QUANT AND CODED VALUES
A40 064B8 00000000 (02423)      * STAB$SZ=PA-STAR$SA JCOMPUTE SIZE
A41 064BA 00000033 (02424)      * END
A42 064BC 00000000 (02425)      *
A43 064BE 00000000 (02426)      *
A44 064C0 00000000 (02427)      *
A45 064C2 00000000 (02428)      *
A46 064C4 00000000 (02429)      *
A47 064C6 00000000 (02430)      *
A48 064C8 00000000 (02431)      *
A49 064CA 00000000 (02432)      *
A50 064CC 00000000 (02433)      *
A51 064CE 00000000 (02434)      *
A52 064D0 00000000 (02435)      *
A53 064D2 00000000 (02436)      *
A54 064D4 00000000 (02437)      *
A55 064D6 00000000 (02438)      *
A56 064D8 00000000 (02439)      *
A57 064DA 00000000 (02440)      *
A58 064DC 00000000 (02441)      *
A59 064DE 00000000 (02442)      *
A60 064E0 00000000 (02443)      *
A61 064E2 00000000 (02444)      *
A62 064E4 00000000 (02445)      *
A63 064E6 00000000 (02446)      *
A64 064E8 00000000 (02447)      *
A65 064EA 00000000 (02448)      *
A66 064EC 00000000 (02449)      *
A67 064EE 00000000 (02450)      *
A68 064F0 00000000 (02451)      *
A69 064F2 00000000 (02452)      *
A70 064F4 00000000 (02453)      *
A71 064F6 00000000 (02454)      *
A72 064F8 00000000 (02455)      *
A73 064FA 00000000 (02456)      *
A74 064FC 00000000 (02457)      *
A75 064FE 00000000 (02458)      *
A76 06500 00000000 (02459)      *
A77 06502 00000000 (02460)      *
A78 06504 00000000 (02461)      *
A79 06506 00000000 (02462)      *
A80 06508 00000000 (02463)      *
A81 0650A 00000000 (02464)      *
A82 0650C 00000000 (02465)      *
A83 0650E 00000000 (02466)      *
A84 06510 00000000 (02467)      *
A85 06512 00000000 (02468)      *
A86 06514 00000000 (02469)      *
A87 06516 00000000 (02470)      *
A88 06518 00000000 (02471)      *
A89 0651A 00000000 (02472)      *
A90 0651C 00000000 (02473)      *
A91 0651E 00000000 (02474)      *
A92 06520 00000000 (02475)      *
A93 06522 00000000 (02476)      *
A94 06524 00000000 (02477)      *
A95 06526 00000000 (02478)      *
A96 06528 00000000 (02479)      *
A97 0652A 00000000 (02480)      *
A98 0652C 00000000 (02481)      *
A99 0652E 00000000 (02482)      *
A00 06530 00000000 (02483)      *
A01 06532 00000000 (02484)      *
A02 06534 00000000 (02485)      *
A03 06536 00000000 (02486)      *
A04 06538 00000000 (02487)      *
A05 0653A 00000000 (02488)      *
A06 0653C 00000000 (02489)      *
A07 0653E 00000000 (02490)      *
A08 06540 00000000 (02491)      *
A09 06542 00000000 (02492)      *
A10 06544 00000000 (02493)      *
A11 06546 00000000 (02494)      *
A12 06548 00000000 (02495)      *
A13 0654A 00000000 (02496)      *
A14 0654C 00000000 (02497)      *
A15 0654E 00000000 (02498)      *
A16 06550 00000000 (02499)      *
A17 06552 00000000 (02500)      *
A18 06554 00000000 (02501)      *
A19 06556 00000000 (02502)      *
A20 06558 00000000 (02503)      *
A21 0655A 00000000 (02504)      *
A22 0655C 00000000 (02505)      *
A23 0655E 00000000 (02506)      *
A24 06560 00000000 (02507)      *
A25 06562 00000000 (02508)      *
A26 06564 00000000 (02509)      *
A27 06566 00000000 (02510)      *
A28 06568 00000000 (02511)      *
A29 0656A 00000000 (02512)      *
A30 0656C 00000000 (02513)      *
A31 0656E 00000000 (02514)      *
A32 06570 00000000 (02515)      *
A33 06572 00000000 (02516)      *
A34 06574 00000000 (02517)      *
A35 06576 00000000 (02518)      *
A36 06578 00000000 (02519)      *
A37 0657A 00000000 (02520)      *
A38 0657C 00000000 (02521)      *
A39 0657E 00000000 (02522)      *
A40 06580 00000000 (02523)      *
A41 06582 00000000 (02524)      *
A42 06584 00000000 (02525)      *
A43 06586 00000000 (02526)      *
A44 06588 00000000 (02527)      *
A45 0658A 00000000 (02528)      *
A46 0658C 00000000 (02529)      *
A47 0658E 00000000 (02530)      *
A48 06590 00000000 (02531)      *
A49 06592 00000000 (02532)      *
A50 06594 00000000 (02533)      *
A51 06596 00000000 (02534)      *
A52 06598 00000000 (02535)      *
A53 0659A 00000000 (02536)      *
A54 0659C 00000000 (02537)      *
A55 0659E 00000000 (02538)      *
A56 065A0 00000000 (02539)      *
A57 065A2 00000000 (02540)      *
A58 065A4 00000000 (02541)      *
A59 065A6 00000000 (02542)      *
A60 065A8 00000000 (02543)      *
A61 065AA 00000000 (02544)      *
A62 065AC 00000000 (02545)      *
A63 065AE 00000000 (02546)      *
A64 065B0 00000000 (02547)      *
A65 065B2 00000000 (02548)      *
A66 065B4 00000000 (02549)      *
A67 065B6 00000000 (02550)      *
A68 065B8 00000000 (02551)      *
A69 065BA 00000000 (02552)      *
A70 065BC 00000000 (02553)      *
A71 065BE 00000000 (02554)      *
A72 065C0 00000000 (02555)      *
A73 065C2 00000000 (02556)      *
A74 065C4 00000000 (02557)      *
A75 065C6 00000000 (02558)      *
A76 065C8 00000000 (02559)      *
A77 065CA 00000000 (02560)      *
A78 065CC 00000000 (02561)      *
A79 065CE 00000000 (02562)      *
A80 065D0 00000000 (02563)      *
A81 065D2 00000000 (02564)      *
A82 065D4 00000000 (02565)      *
A83 065D6 00000000 (02566)      *
A84 065D8 00000000 (02567)      *
A85 065DA 00000000 (02568)      *
A86 065DC 00000000 (02569)      *
A87 065DE 00000000 (02570)      *
A88 065E0 00000000 (02571)      *
A89 065E2 00000000 (02572)      *
A90 065E4 00000000 (02573)      *
A91 065E6 00000000 (02574)      *
A92 065E8 00000000 (02575)      *
A93 065EA 00000000 (02576)      *
A94 065EC 00000000 (02577)      *
A95 065EE 00000000 (02578)      *
A96 065F0 00000000 (02579)      *
A97 065F2 00000000 (02580)      *
A98 065F4 00000000 (02581)      *
A99 065F6 00000000 (02582)      *
A00 065F8 00000000 (02583)      *
A01 065FA 00000000 (02584)      *
A02 065FC 00000000 (02585)      *
A03 065FE 00000000 (02586)      *
A04 06600 00000000 (02587)      *
A05 06602 00000000 (02588)      *
A06 06604 00000000 (02589)      *
A07 06606 00000000 (02590)      *
A08 06608 00000000 (02591)      *
A09 0660A 00000000 (02592)      *
A10 0660C 00000000 (02593)      *
A11 0660E 00000000 (02594)      *
A12 06610 00000000 (02595)      *
A13 06612 00000000 (02596)      *
A14 06614 00000000 (02597)      *
A15 06616 00000000 (02598)      *
A16 06618 00000000 (02599)      *
A17 0661A 00000000 (02600)      *
A18 0661C 00000000 (02601)      *
A19 0661E 00000000 (02602)      *
A20 06620 00000000 (02603)      *
A21 06622 00000000 (02604)      *
A22 06624 00000000 (02605)      *
A23 06626 00000000 (02606)      *
A24 06628 00000000 (02607)      *
A25 0662A 00000000 (02608)      *
A26 0662C 00000000 (02609)      *
A27 0662E 00000000 (02610)      *
A28 06630 00000000 (02611)      *
A29 06632 00000000 (02612)      *
A30 06634 00000000 (02613)      *
A31 06636 00000000 (02614)      *
A32 06638 00000000 (02615)      *
A33 0663A 00000000 (02616)      *
A34 0663C 00000000 (02617)      *
A35 0663E 00000000 (02618)      *
A36 06640 00000000 (02619)      *
A37 06642 00000000 (02620)      *
A38 06644 00000000 (02621)      *
A39 06646 00000000 (02622)      *
A40 06648 00000000 (02623)      *
A41 0664A 00000000 (02624)      *
A42 0664C 00000000 (02625)      *
A43 0664E 00000000 (02626)      *
A44 06650 00000000 (02627)      *
A45 06652 00000000 (02628)      *
A46 06654 00000000 (02629)      *
A47 06656 00000000 (02630)      *
A48 06658 00000000 (02631)      *
A49 0665A 00000000 (02632)      *
A50 0665C 00000000 (02633)      *
A51 0665E 00000000 (02634)      *
A52 06660 00000000 (02635)      *
A53 06662 00000000 (02636)      *
A54 06664 00000000 (02637)      *
A55 06666 00000000 (02638)      *
A56 06668 00000000 (02639)      *
A57 0666A 00000000 (02640)      *
A58 0666C 00000000 (02641)      *
A59 0666E 00000000 (02642)      *
A60 06670 00000000 (02643)      *
A61 06672 00000000 (02644)      *
A62 06674 00000000 (02645)      *
A63 06676 00000000 (02646)      *
A64 06678 00000000 (02647)      *
A65 0667A 00000000 (02648)      *
A66 0667C 00000000 (02649)      *
A67 0667E 00000000 (02650)      *
A68 06680 00000000 (02651)      *
A69 06682 00000000 (02652)      *
A70 06684 00000000 (02653)      *
A71 06686 00000000 (02654)      *
A72 06688 00000000 (02655)      *
A73 0668A 00000000 (02656)      *
A74 0668C 00000000 (02657)      *
A75 0668E 00000000 (02658)      *
A76 06690 00000000 (02659)      *
A77 06692 00000000 (02660)      *
A78 06694 00000000 (02661)      *
A79 06696 00000000 (02662)      *
A80 06698 00000000 (02663)      *
A81 0669A 00000000 (02664)      *
A82 0669C 00000000 (02665)      *
A83 0669E 00000000 (02666)      *
A84 066A0 00000000 (02667)      *
A85 066A2 00000000 (02668)      *
A86 066A4 00000000 (02669)      *
A87 066A6 00000000 (02670)      *
A88 066A8 00000000 (02671)      *
A89 066AA 00000000 (02672)      *
A90 066AC 00000000 (02673)      *
A91 066AE 00000000 (02674)      *
A92 066B0 00000000 (02675)      *
A93 066B2 00000000 (02676)      *
A94 066B4 00000000 (02677)      *
A95 066B6 00000000 (02678)      *
A96 066B8 00000000 (02679)      *
A97 066BA 00000000 (02680)      *
A98 066BC 00000000 (02681)      *
A99 066BE 00000000 (02682)      *
A00 066C0 00000000 (02683)      *
A01 066C2 00000000 (02684)      *
A02 066C4 00000000 (02685)      *
A03 066C6 00000000 (02686)      *
A04 066C8 00000000 (02687)      *
A05 066CA 00000000 (02688)      *
A06 066CC 00000000 (02689)      *
A07 066CE 00000000 (02690)      *
A08 066D0 00000000 (02691)      *
A09 066D2 00000000 (02692)      *
A10 066D4 00000000 (02693)      *
A11 066D6 00000000 (02694)      *
A12 066D8 00000000 (02695)      *
A13 066DA 00000000 (02696)      *
A14 066DC 00000000 (02697)      *
A15 066DE 00000000 (02698)      *
A16 066E0 00000000 (02699)      *
A17 066E2 00000000 (02700)      *
A18 066E4 00000000 (02701)      *
A19 066E6 00000000 (02702)      *
A20 066E8 00000000 (02703)      *
A21 066EA 00000000 (02704)      *
A22 066EC 00000000 (02705)      *
A23 066EE 00000000 (02706)      *
A24 066F0 00000000 (02707)      *
A25 066F2 00000000 (02708)      *
A26 066F4 00000000 (02709)      *
A27 066F6 00000000 (02710)      *
A28 066F8 00000000 (02711)      *
A29 066FA 00000000 (02712)      *
A30 066FC 00000000 (02713)      *
A31 066FE 00000000 (02714)      *
A32 06700 00000000 (02715)      *
A33 06702 00000000 (02716)      *
A34 06704 00000000 (02717)      *
A35 06706 00000000 (02718)      *
A36 06708 00000000 (02719)      *
A37 0670A 00000000 (02720)      *
A38 0670C 00000000 (02721)      *
A39 0670E 00000000 (02722)      *
A40 06710 00000000 (02723)      *
A41 06712 00000000 (02724)      *
A42 06714 00000000 (02725)      *
A43 06716 00000000 (02726)      *
A44 06718 00000000 (02727)      *
A45 0671A 00000000 (02728)      *
A46 0671C 00000000 (02729)      *
A47 0671E 00000000 (02730)      *
A48 06720 00000000 (02731)      *
A49 06722 00000000 (02732)      *
A50 06724 00000000 (02733)      *
A51 06726 00000000 (02734)      *
A52 06728 00000000 (02735)      *
A53 0672A 00000000 (02736)      *
A54 0672C 00000000 (02737)      *
A55 0672E 00000000 (02738)      *
A56 06730 00000000 (02739)      *
A57 06732 00000000 (02740)      *
A58 06734 00000000 (02741)      *
A59 06736 00000000 (02742)      *
A60 06738 00000000 (02743)      *
A61 0673A 00000000 (02744)      *
A62 0673C 00000000 (02745)      *
A63 0673E 00000000 (02746)      *
A64 06740 00000000 (02747)      *
A65 06742 00000000 (02748)      *
A66 06744 00000000 (02749)      *
A67 06746 00000000 (02750)      *
A68 06748 00000000 (02751)      *
A69 0674A 00000000 (02752)      *
A70 0674C 00000000 (02753)      *
A71 0674E 00000000 (02754)      *
A72 06750 00000000 (02755)      *
A73 06752 00000000 (02756)      *
A74 06754 00000000 (02757)      *
A75 06756 00000000 (02758)      *
A76 06758 00000000 (02759)      *
A77 0675A 00000000 (02760)      *
A78 0675C 00000000 (02761)      *
A79 0675E 00000000 (02762)      *
A80 06760 00000000 (02763)      *
A81 06762 00000000 (02764)      *
A82 06764 00000000 (02765)      *
A83 06766 00000000 (02766)      *
A84 06768 00000000 (02767)      *
A85 0676A 00000000 (02768)      *
A86 0676C 00000000 (02769)      *
A87 0676E 00000000 (02770)      *
A88 06770 00000000 (02771)      *
A89 06772 00000000 (02772)      *
A90 06774 00000000 (02773)      *
A91 06776 00000000 (02774)      *
A92 06778 00000000 (02775)      *
A93 0677A 00000000 (02776)      *
A94 0677C 00000000 (02777)      *
A95 0677E 00000000 (02778)      *
A96 06780 00000000 (02779)      *
A97 06782 00000000 (02780)      *
A98 06784 00000000 (02781)      *
A99 06786 00000000 (02782)      *
A00 06788 00000000 (02783)      *
A01 0678A 00000000 (02784)      *
A02 0678C 00000000 (02785)      *
A03 0678E 00000000 (02786)      *
A04 06790 00000000 (02787)      *
A05 06792 00000000 (02788)      *
A06 06794 00000000 (02789)      *
A07 06796 00000000 (02790)      *
A08 06798 00000000 (02791)      *
A09 0679A 00000000 (02792)      *
A10 0679C 00000000 (02793)      *
A11 0679E 00000000 (02794)      *
A12 067A0 00000000 (02795)      *
A13 067A2 00000000 (02796)      *
A14 067A4 00000000 (02797)      *
A15 067A6 00000000 (02798)      *
A16 067A8 00000000 (02799)      *
A17 067AA 00000000 (02800)      *
A18 067AC 00000000 (02801)      *
A19 067AE 00000000 (02802)      *
A20 067B0 00000000 (02803)      *
A21 067B2 00000000 (02804)      *
A22 067B4 00000000 (02805)      *
A23 067B6 00000000 (02806)      *
A24 067B8 00000000 (02807)      *
A25 067BA 00000000 (02808)      *
A26 067BC 00000000 (02809)      *
A27 067BE 00000000 (02810)      *
A28 067C0 00000000 (02811)      *
A29 067C2 00000000 (02812)      *
A30 067C4 00000000 (
```

195

PAGE 58: (00001)C0C16>B0016M,MSU.2, 29-Dec-80 14:30:40, Ed: WOLF
AP03-INVP(V,A,U,V) 3-TAP PITCH INVERSE FILTER

```

(02539)  AP03-INVP(V,A,U,V)  3-TAP PITCH INVERSE FILTER
(02540)  J. WOLF, 6/30/80
(02541)  )
(02542)  ) BINDS TO AP03-INVP
(02543)  )
(02544)  )
(02545)  ) IMPLEMENTS A 3-TAP PITCH INVERSE FILTER
(02546)  )  $Y(N) = U(N) + V(0) \cdot U(N-M+1) + V(1) \cdot U(N-M) + V(2) \cdot U(N-M-1)$ 
(02547)  )  $N=0,1,...,VBS-1$ 
(02548)  )
(02549)  ) BUFFER V = FILTER OUTPUT
(02550)  ) REAL SCALAN A = PITCH PERIOD IN SAMPLES
(02551)  ) BUFFER U = CURRENT FILTER INPUT AND PREVIOUS 216 SAMPLES OF INPUT,
(02552)  ) WHERE THE INDEX FOR U RANGES FROM -216 TO VBS-1. THAT IS, THE
(02553)  ) FIRST ENTRY IN U IS "LOGICAL" U(-216). BUFFER U SHOULD BE
(02554)  ) (AT LEAST) 216+VBS SAMPLES LONG.
(02555)  ) NOTE: BUFFER U MUST BE COMPACT 32-BIT FLT PT
(02556)  ) BUFFER V = FILTER COEFFICIENTS
(02557)  )
(02558)  ) WE PERFORM 2 FILTER CALCULATIONS IN THE 2 ADMS:
(02559)  ) LEFT:
(02560)  )  $V(N) = ((V(0) \cdot U(N-M+1) + U(N)) + V(1) \cdot U(N-M)) + V(2) \cdot U(N-M-1)$ 
(02561)  ) RIGHT:
(02562)  )  $Y(N+1) = ((V(1) \cdot U(N-M+1) + U(N-M+1)) + V(2) \cdot U(N-M)) + V(0) \cdot U(N-M-2)$ 
(02563)  )
(02564)  ) INPUT SEQUENCE:
(02565)  ) (2*-15), SA, MOD. APS INSTR., V(0), V(1), V(2),
(02566)  ) U(-M+1), U(0), U(1), U(-M), U(-M-1), U(-M+2),
(02567)  ) U(2-M+1), U(2), U(3), U(2-M), U(2-M+2), ...,
(02568)  ) U(M-M+1), U(M), U(M+1), U(M-M), U(M-M-1), U(M-M+2), ..., [PI]
(02569)  ) OUTPUT SEQUENCE:
(02570)  ) FPD PT PITCH, MOD. APS INSTR., 2 DUMMIES, V(1), V(2), ..., Y(UBS) [EOI]
(02571)  )
(02572)  ) DURATION:
(02573)  ) 18 APS CYCLES = "1.8 USEC PER 2 OUTPUT POINTS
(02574)  ) 8 MEM REFS = "1.44 USEC PER 2 OUTPUT POINTS (BUS 3)
(02575)  )
(02576)  ) APU REGISTER USAGE:
(02577)  ) LEFT
(02578)  ) A0 SUM
(02579)  ) A1 V*V
(02580)  ) A2 U(N)
(02581)  ) M0 U(N-M+1)
(02582)  ) M1 U(N-M)
(02583)  ) M2 U(N-M-1)
(02584)  ) M4 V(0)
(02585)  ) M5 V(1)
(02586)  ) M6 V(2)
(02587)  )
(02588)  ) EVEN
(02589)  ) DATA INVPUS$A
(02590)  ) DATA INVPUS$Z
(02591)  )
(02592)  )
(02593)  )
(02594)  )
(02595)  )
(02596)  )
(02597)  )
(02598)  )
(02599)  )
(02600)  )
(02601)  )
(02602)  )
(02603)  )
(02604)  )
(02605)  )
(02606)  )
(02607)  )
(02608)  )
(02609)  )
(02610)  )
(02611)  )
(02612)  )
(02613)  )
(02614)  )
(02615)  )
(02616)  )
(02617)  )
(02618)  )
(02619)  )
(02620)  )
(02621)  )
(02622)  )
(02623)  )
(02624)  )
(02625)  )
(02626)  )
(02627)  )
(02628)  )
(02629)  )
(02630)  )
(02631)  )
(02632)  )
(02633)  )
(02634)  )
(02635)  )
(02636)  )
(02637)  )
(02638)  )
(02639)  )
(02640)  )
(02641)  )
(02642)  )
(02643)  )
(02644)  )
(02645)  )
(02646)  )
(02647)  )
(02648)  )
(02649)  )
(02650)  )
(02651)  )
(02652)  )
(02653)  )
(02654)  )
(02655)  )
(02656)  )
(02657)  )
(02658)  )
(02659)  )
(02660)  )
(02661)  )
(02662)  )
(02663)  )
(02664)  )
(02665)  )
(02666)  )
(02667)  )
(02668)  )
(02669)  )
(02670)  )
(02671)  )
(02672)  )
(02673)  )
(02674)  )
(02675)  )
(02676)  )
(02677)  )
(02678)  )
(02679)  )
(02680)  )
(02681)  )
(02682)  )
(02683)  )
(02684)  )
(02685)  )
(02686)  )
(02687)  )
(02688)  )
(02689)  )
(02690)  )
(02691)  )
(02692)  )
(02693)  )
(02694)  )
(02695)  )
(02696)  )
(02697)  )
(02698)  )
(02699)  )
(02700)  )
(02701)  )
(02702)  )
(02703)  )
(02704)  )
(02705)  )
(02706)  )
(02707)  )
(02708)  )
(02709)  )
(02710)  )
(02711)  )
(02712)  )
(02713)  )
(02714)  )
(02715)  )
(02716)  )
(02717)  )
(02718)  )
(02719)  )
(02720)  )
(02721)  )
(02722)  )
(02723)  )
(02724)  )
(02725)  )
(02726)  )
(02727)  )
(02728)  )
(02729)  )
(02730)  )
(02731)  )
(02732)  )
(02733)  )
(02734)  )
(02735)  )
(02736)  )
(02737)  )
(02738)  )
(02739)  )
(02740)  )
(02741)  )
(02742)  )
(02743)  )
(02744)  )
(02745)  )
(02746)  )
(02747)  )
(02748)  )
(02749)  )
(02750)  )
(02751)  )
(02752)  )
(02753)  )
(02754)  )
(02755)  )
(02756)  )
(02757)  )
(02758)  )
(02759)  )
(02760)  )
(02761)  )
(02762)  )
(02763)  )
(02764)  )
(02765)  )
(02766)  )
(02767)  )
(02768)  )
(02769)  )
(02770)  )
(02771)  )
(02772)  )
(02773)  )
(02774)  )
(02775)  )
(02776)  )
(02777)  )
(02778)  )
(02779)  )
(02780)  )
(02781)  )
(02782)  )
(02783)  )
(02784)  )
(02785)  )
(02786)  )
(02787)  )
(02788)  )
(02789)  )
(02790)  )
(02791)  )
(02792)  )
(02793)  )
(02794)  )
(02795)  )
(02796)  )
(02797)  )
(02798)  )
(02799)  )
(02800)  )
(02801)  )
(02802)  )
(02803)  )
(02804)  )
(02805)  )
(02806)  )
(02807)  )
(02808)  )
(02809)  )
(02810)  )
(02811)  )
(02812)  )
(02813)  )
(02814)  )
(02815)  )
(02816)  )
(02817)  )
(02818)  )
(02819)  )
(02820)  )
(02821)  )
(02822)  )
(02823)  )
(02824)  )
(02825)  )
(02826)  )
(02827)  )
(02828)  )
(02829)  )
(02830)  )
(02831)  )
(02832)  )
(02833)  )
(02834)  )
(02835)  )
(02836)  )
(02837)  )
(02838)  )
(02839)  )
(02840)  )
(02841)  )
(02842)  )
(02843)  )
(02844)  )
(02845)  )
(02846)  )
(02847)  )
(02848)  )
(02849)  )
(02850)  )
(02851)  )
(02852)  )
(02853)  )
(02854)  )
(02855)  )
(02856)  )
(02857)  )
(02858)  )
(02859)  )
(02860)  )
(02861)  )
(02862)  )
(02863)  )
(02864)  )
(02865)  )
(02866)  )
(02867)  )
(02868)  )
(02869)  )
(02870)  )
(02871)  )
(02872)  )
(02873)  )
(02874)  )
(02875)  )
(02876)  )
(02877)  )
(02878)  )
(02879)  )
(02880)  )
(02881)  )
(02882)  )
(02883)  )
(02884)  )
(02885)  )
(02886)  )
(02887)  )
(02888)  )
(02889)  )
(02890)  )
(02891)  )
(02892)  )
(02893)  )
(02894)  )
(02895)  )
(02896)  )
(02897)  )
(02898)  )
(02899)  )
(02900)  )
(02901)  )
(02902)  )
(02903)  )
(02904)  )
(02905)  )
(02906)  )
(02907)  )
(02908)  )
(02909)  )
(02910)  )
(02911)  )
(02912)  )
(02913)  )
(02914)  )
(02915)  )
(02916)  )
(02917)  )
(02918)  )
(02919)  )
(02920)  )
(02921)  )
(02922)  )
(02923)  )
(02924)  )
(02925)  )
(02926)  )
(02927)  )
(02928)  )
(02929)  )
(02930)  )
(02931)  )
(02932)  )
(02933)  )
(02934)  )
(02935)  )
(02936)  )
(02937)  )
(02938)  )
(02939)  )
(02940)  )
(02941)  )
(02942)  )
(02943)  )
(02944)  )
(02945)  )
(02946)  )
(02947)  )
(02948)  )
(02949)  )
(02950)  )
(02951)  )
(02952)  )
(02953)  )
(02954)  )
(02955)  )
(02956)  )
(02957)  )
(02958)  )
(02959)  )
(02960)  )
(02961)  )
(02962)  )
(02963)  )
(02964)  )
(02965)  )
(02966)  )
(02967)  )
(02968)  )
(02969)  )
(02970)  )
(02971)  )
(02972)  )
(02973)  )
(02974)  )
(02975)  )
(02976)  )
(02977)  )
(02978)  )
(02979)  )
(02980)  )
(02981)  )
(02982)  )
(02983)  )
(02984)  )
(02985)  )
(02986)  )
(02987)  )
(02988)  )
(02989)  )
(02990)  )
(02991)  )
(02992)  )
(02993)  )
(02994)  )
(02995)  )
(02996)  )
(02997)  )
(02998)  )
(02999)  )
(03000)  )
(03001)  )
(03002)  )
(03003)  )
(03004)  )
(03005)  )
(03006)  )
(03007)  )
(03008)  )
(03009)  )
(03010)  )
(03011)  )
(03012)  )
(03013)  )
(03014)  )
(03015)  )
(03016)  )
(03017)  )
(03018)  )
(03019)  )
(03020)  )
(03021)  )
(03022)  )
(03023)  )
(03024)  )
(03025)  )
(03026)  )
(03027)  )
(03028)  )
(03029)  )
(03030)  )
(03031)  )
(03032)  )
(03033)  )
(03034)  )
(03035)  )
(03036)  )
(03037)  )
(03038)  )
(03039)  )
(03040)  )
(03041)  )
(03042)  )
(03043)  )
(03044)  )
(03045)  )
(03046)  )
(03047)  )
(03048)  )
(03049)  )
(03050)  )
(03051)  )
(03052)  )
(03053)  )
(03054)  )
(03055)  )
(03056)  )
(03057)  )
(03058)  )
(03059)  )
(03060)  )
(03061)  )
(03062)  )
(03063)  )
(03064)  )
(03065)  )
(03066)  )
(03067)  )
(03068)  )
(03069)  )
(03070)  )
(03071)  )
(03072)  )
(03073)  )
(03074)  )
(03075)  )
(03076)  )
(03077)  )
(03078)  )
(03079)  )
(03080)  )
(03081)  )
(03082)  )
(03083)  )
(03084)  )
(03085)  )
(03086)  )
(03087)  )
(03088)  )
(03089)  )
(03090)  )
(03091)  )
(03092)  )
(03093)  )
(03094)  )
(03095)  )
(03096)  )
(03097)  )
(03098)  )
(03099)  )
(03100)  )
(03101)  )
(03102)  )
(03103)  )
(03104)  )
(03105)  )
(03106)  )
(03107)  )
(03108)  )
(03109)  )
(03110)  )
(03111)  )
(03112)  )
(03113)  )
(03114)  )
(03115)  )
(03116)  )
(03117)  )
(03118)  )
(03119)  )
(03120)  )
(03121)  )
(03122)  )
(03123)  )
(03124)  )
(03125)  )
(03126)  )
(03127)  )
(03128)  )
(03129)  )
(03130)  )
(03131)  )
(03132)  )
(03133)  )
(03134)  )
(03135)  )
(03136)  )
(03137)  )
(03138)  )
(03139)  )
(03140)  )
(03141)  )
(03142)  )
(03143)  )
(03144)  )
(03145)  )
(03146)  )
(03147)  )
(03148)  )
(03149)  )
(03150)  )
(03151)  )
(03152)  )
(03153)  )
(03154)  )
(03155)  )
(03156)  )
(03157)  )
(03158)  )
(03159)  )
(03160)  )
(03161)  )
(03162)  )
(03163)  )
(03164)  )
(03165)  )
(03166)  )
(03167)  )
(03168)  )
(03169)  )
(03170)  )
(03171)  )
(03172)  )
(03173)  )
(03174)  )
(03175)  )
(03176)  )
(03177)  )
(03178)  )
(03179)  )
(03180)  )
(03181)  )
(03182)  )
(03183)  )
(03184)  )
(03185)  )
(03186)  )
(03187)  )
(03188)  )
(03189)  )
(03190)  )
(03191)  )
(03192)  )
(03193)  )
(03194)  )
(03195)  )
(03196)  )
(03197)  )
(03198)  )
(03199)  )
(03200)  )
(03201)  )
(03202)  )
(03203)  )
(03204)  )
(03205)  )
(03206)  )
(03207)  )
(03208)  )
(03209)  )
(03210)  )
(03211)  )
(03212)  )
(03213)  )
(03214)  )
(03215)  )
(03216)  )
(03217)  )
(03218)  )
(03219)  )
(03220)  )
(03221)  )
(03222)  )
(03223)  )
(03224)  )
(03225)  )
(03226)  )
(03227)  )
(03228)  )
(03229)  )
(03230)  )
(03231)  )
(03232)  )
(03233)  )
(03234)  )
(03235)  )
(03236)  )
(03237)  )
(03238)  )
(03239)  )
(03240)  )
(03241)  )
(03242)  )
(03243)  )
(03244)  )
(03245)  )
(03246)  )
(03247)  )
(03248)  )
(03249)  )
(03250)  )
(03251)  )
(03252)  )
(03253)  )
(03254)  )
(03255)  )
(03256)  )
(03257)  )
(03258)  )
(03259)  )
(03260)  )
(03261)  )
(03262)  )
(03263)  )
(03264)  )
(03265)  )
(03266)  )
(03267)  )
(03268)  )
(03269)  )
(03270)  )
(03271)  )
(03272)  )
(03273)  )
(03274)  )
(03275)  )
(03276)  )
(03277)  )
(03278)  )
(03279)  )
(03280)  )
(03281)  )
(03282)  )
(03283)  )
(03284)  )
(03285)  )
(03286)  )
(03287)  )
(03288)  )
(03289)  )
(03290)  )
(03291)  )
(03292)  )
(03293)  )
(03294)  )
(03295)  )
(03296)  )
(03297)  )
(03298)  )
(03299)  )
(03300)  )
(03301)  )
(03302)  )
(03303)  )
(03304)  )
(03305)  )
(03306)  )
(03307)  )
(03308)  )
(03309)  )
(03310)  )
(03311)  )
(03312)  )
(03313)  )
(03314)  )
(03315)  )
(03316)  )
(03317)  )
(03318)  )
(03319)  )
(03320)  )
(03321)  )
(03322)  )
(03323)  )
(03324)  )
(03325)  )
(03326)  )
(03327)  )
(03328)  )
(03329)  )
(03330)  )
(03331)  )
(03332)  )
(03333)  )
(03334)  )
(03335)  )
(03336)  )
(03337)  )
(03338)  )
(03339)  )
(03340)  )
(03341)  )
(03342)  )
(03343)  )
(03344)  )
(03345)  )
(03346)  )
(03347)  )
(03348)  )
(03349)  )
(03350)  )
(03351)  )
(03352)  )
(03353)  )
(03354)  )
(03355)  )
(03356)  )
(03357)  )
(03358)  )
(03359)  )
(03360)  )
(03361)  )
(03362)  )
(03363)  )
(03364)  )
(03365)  )
(03366)  )
(03367)  )
(03368)  )
(03369)  )
(03370)  )
(03371)  )
(03372)  )
(03373)  )
(03374)  )
(03375)  )
(03376)  )
(03377)  )
(03378)  )
(03379)  )
(03380)  )
(03381)  )
(03382)  )
(03383)  )
(03384)  )
(03385)  )
(03386)  )
(03387)  )
(03388)  )
(03389)  )
(03390)  )
(03391)  )
(03392)  )
(03393)  )
(03394)  )
(03395)  )
(03396)  )
(03397)  )
(03398)  )
(03399)  )
(03400)  )
(03401)  )
(03402)  )
(03403)  )
(03404)  )
(03405)  )
(03406)  )
(03407)  )
(03408)  )
(03409)  )
(03410)  )
(03411)  )
(03412)  )
(03413)  )
(03414)  )
(03415)  )
(03416)  )
(03417)  )
(03418)  )
(03419)  )
(03420)  )
(03421)  )
(03422)  )
(03423)  )
(03424)  )
(03425)  )
(03426)  )
(03427)  )
(03428)  )
(03429)  )
(03430)  )
(03431)  )
(03432)  )
(03433)  )
(03434)  )
(03435)  )
(03436)  )
(03437)  )
(03438)  )
(03439)  )
(03440)  )
(03441)  )
(03442)  )
(03443)  )
(03444)  )
(03445)  )
(03446)  )
(03447)  )
(03448)  )
(03449)  )
(03450)  )
(03451)  )
(03452)  )
(03453)  )
(03454)  )
(03455)  )
(03456)  )
(03457)  )
(03458)  )
(03459)  )
(03460)  )
(03461)  )
(03462)  )
(03463)  )
(03464)  )
(03465)  )
(03466)  )
(03467)  )
(03468)  )
(03469)  )
(03470)  )
(03471)  )
(03472)  )
(03473)  )
(03474)  )
(03475)  )
(03476)  )
(03477)  )
(03478)  )
(03479)  )
(03480)  )
(03481)  )
(03482)  )
(03483)  )
(03484)  )
(03485)  )
(03486)  )
(03487)  )
(03488)  )
(03489)  )
(03490)  )
(03491)  )
(03492)  )
(03493)  )
(03494)  )
(03495)  )
(03496)  )
(03497)  )
(03498)  )
(03499)  )
(03500)  )
(03501)  )
(03502)  )
(03503)  )
(03504)  )
(03505)  )
(03506)  )
(03507)  )
(03508)  )
(03509)  )
(03510)  )
(03511)  )
(03512)  )
(03513)  )
(03514)  )
(03515)  )
(03516)  )
(03517)  )
(03518)  )
(03519)  )
(03520)  )
(03521)  )
(03522)  )
(03523)  )
(03524)  )
(03525)  )
(03526)  )
(03527)  )
(03528)  )
(03529)  )
(03530)  )
(03531)  )
(03532)  )
(03533)  )
(03534)  )
(03535)  )
(03536)  )
(03537)  )
(03538)  )
(03539)  )
(03540)  )
(03541)  )
(03542)  )
(03543)  )
(03544)  )
(03545)  )
(03546)  )
(03547)  )
(03548)  )
(03549)  )
(03550)  )
(03551)  )
(03552)  )
(03553)  )
(03554)  )
(03555)  )
(03556)  )
(03557)  )
(03558)  )
(03559)  )
(03560)  )
(03561)  )
(03562)  )
(03563)  )
(03564)  )
(03565)  )
(03566)  )
(03567)  )
(03568)  )
(03569)  )
(03570)  )
(03571)  )
(03572)  )
(03573)  )
(03574)  )
(03575)  )
(03576)  )
(03577)  )
(03578)  )
(03579)  )
(03580)  )
(03581)  )
(03582)  )
(03583)  )
(03584)  )
(03585)  )
(03586)  )
(03587)  )
(03588)  )
(03589)  )
(03590)  )
(03591)  )
(03592)  )
(03593)  )
(03594)  )
(03595)  )
(03596)  )
(03597)  )
(03598)  )
(03599)  )
(03600)  )
(03601)  )
(03602)  )
(03603)  )
(03604)  )
(03605)  )
(03606)  )
(03607)  )
(03608)  )
(03609)  )
(03610)  )
(03611)  )
(03612)  )
(03613)  )
(03614)  )
(03615)  )
(03616)  )
(03617)  )
(03618)  )
(03619)  )
(03620)  )
(03621)  )
(03622)  )
(03623)  )
(03624)  )
(03625)  )
(03626)  )
(03627)  )
(03628)  )
(03629)  )
(03630)  )
(03631)  )
(03632)  )
(03633)  )
(03634)  )
(03635)  )
(03636)  )
(03637)  )
(03638)  )
(03639)  )
(03640)  )
(03641)  )
(03642)  )
(03643)  )
(03644)  )
(03645)  )
(03646)  )
(03647)  )
(03648)  )
(03649)  )
(03650)  )
(03651)  )
(03652)  )
(03653)  )
(03654)  )
(03655)  )
(03656)  )
(03657)  )
(03658)  )
(03659)  )
(03660)  )
(03661)  )
(03662)  )
(03663)  )
(03664)  )
(03665)  )
(03666)  )
(03667)  )
(03668)  )
(03669)  )
(03670)  )
(03671)  )
(03672)  )
(03673)  )
(03674)  )
(03675)  )
(03676)  )
(03677)  )
(03678)  )
(03679)  )
(03680)  )
(03681)  )
(03682)  )
(03683)  )
(03684)  )
(03685)  )
(03686)  )
(03687)  )
(03688)  )
(03689)  )
(03690)  )
(03691)  )
(03692)  )
(03693)  )
(03694)  )
(03695)  )
(03696)  )
(03697)  )
(03698)  )
(03699) 
```

PAGE 59: [RNDJ]DCAL16>DMM16.NSO.2, 29-Dec-88 14:38:40, Ed: WOLF
APU3-INVPF(V,A,U,V) 3-TAP PITCH INVERSE FILTER

```

A00 0651A 00200000 (02592) INVPUS: BEGIN APU(INVPF)
A01 0651C 00EC0000 (02593) INVPUS$A: MOV(IQA,M0) \ NOP
A02 0651E 04000000 (02594) MOV(IQA,M4) \ NOP
A03 06520 00800000 (02595) MUL(M0,M4) \ NOP
A04 06522 3A000000 (02596) MOV(P,A0) \ NOP
A05 06524 009C0000 (02597) ALIGN(A0) \ NOP
A06 06526 90100006 (02598) MOV(R,DQ) \ NOP
A07 06528 00000000 (02599) WHERE WE WAIT ON QOE, PROCEED APS, READ/WRITE THE INSTRUCTION, WAIT
A08 0652A 20372037 (02600) 3 AND PROCEED THE APS AGAIN, AND THEN GET DOWN TO THE COMPUTATION.
A09 0652C 00FC0000 (02601) JUMPC(B1,ONE)
A10 0652E 9010000A (02602) NOP
A11 06530 00000000 (02603) CLEAR(VI)
A12 06532 20372037 (02604) MOV(IQA,DQ) \ NOP
A13 06534 00EC0000 (02605) JUMPC(B2,QOE)
A14 06536 00E00000 (02606) NOP
A15 06538 00E00000 (02607) CLEAR(VI)
A16 0653A 00EC0000 (02608) MOV(IQA,M4) \ MOV(IQA,M6)
A17 0653C 00E00000 (02609) MOV(IQA,M5) \ MOV(IQA,M4)
A18 0653E 00E00000 (02610) MOV(IQA,M6) \ MOV(IQA,M5)
A19 06540 00E00000 (02611) JMAIN LOOP
A20 06542 00000000 (02612) MOV(IQA,M0)
A21 06544 009C0000 (02613) MOV(A1),MUL(M0,M4)
A22 06546 00E00000 (02614) MOV(A0),ADD(A0,A1)
A23 06548 00F20000 (02615) MOV(IQA,A2) \ NOP
A24 0654A 00000000 (02616) NOP \ MOV(IQA,A2)
A25 0654C 009C0000 (02617) MOV(R,DQ)
A26 0654E 00E00000 (02618) MOV(IQA,M1)
A27 06550 00E00000 (02619) MOV(A1),MUL(M1,M5)
A28 06552 00E00000 (02620) ADD(A2,A1)
A29 06554 00E00000 (02621) MOV(IQA,M2) \ NOP
A30 06556 00000000 (02622) NOP \ MOV(IQA,M2)
A31 06558 00510551 (02623) MOV(A1),MUL(M2,M6)
A32 0655A 00E00000 (02624) MOV(A0),ADD(A0,A1)
A33 0655C 00E00000 (02625) JUMPC(B3,E0)
A34 0655E 20372037 (02626) CLEAR(A0)
A35 06560 00000000 (02627) JUMP(0)
A36 06562 00000000 (02628) INVPUS$Z-B1-INVPUS$A
A37 06564 00000000 (02629) END
A38 06566 00000000 (02630)
A39 06568 00000000 (02631)
A40 0656A 00000000 (02632)
A41 0656C 00000000 (02633)
A42 0656E 00000000 (02634)
A43 06570 00000000 (02635)
A44 06572 00000000 (02636)
A45 06574 00000000 (02637)
A46 06576 00000000 (02638)
A47 06578 00000000 (02639)
A48 0657A 00000000 (02640)
A49 0657C 00000000 (02641)
A50 0657E 00000000 (02642)
A51 06580 00000000 (02643)
A52 06582 00000000 (02644)
A53 06584 00000000 (02645)
A54 06586 00000000 (02646)
A55 06588 00000000 (02647)
A56 0658A 00000000 (02648)
A57 0658C 00000000 (02649)
A58 0658E 00000000 (02650)
A59 06590 00000000 (02651)
A60 06592 00000000 (02652)
A61 06594 00000000 (02653)
A62 06596 00000000 (02654)
A63 06598 00000000 (02655)
A64 0659A 00000000 (02656)
A65 0659C 00000000 (02657)
A66 0659E 00000000 (02658)
A67 065A0 00000000 (02659)
A68 065A2 00000000 (02660)
A69 065A4 00000000 (02661)
A70 065A6 00000000 (02662)
A71 065A8 00000000 (02663)
A72 065AA 00000000 (02664)
A73 065AC 00000000 (02665)
A74 065AE 00000000 (02666)
A75 065B0 00000000 (02667)
A76 065B2 00000000 (02668)
A77 065B4 00000000 (02669)
A78 065B6 00000000 (02670)
A79 065B8 00000000 (02671)
A80 065BA 00000000 (02672)
A81 065BC 00000000 (02673)
A82 065BE 00000000 (02674)
A83 065C0 00000000 (02675)
A84 065C2 00000000 (02676)
A85 065C4 00000000 (02677)
A86 065C6 00000000 (02678)
A87 065C8 00000000 (02679)
A88 065CA 00000000 (02680)
A89 065CC 00000000 (02681)
A90 065CE 00000000 (02682)
A91 065C0 00000000 (02683)
A92 065D0 00000000 (02684)
A93 065D2 00000000 (02685)
A94 065D4 00000000 (02686)
A95 065D6 00000000 (02687)
A96 065D8 00000000 (02688)
A97 065DA 00000000 (02689)
A98 065DC 00000000 (02690)
A99 065DE 00000000 (02691)
A00 065E0 00000000 (02692)
A01 065E2 00000000 (02693)
A02 065E4 00000000 (02694)
A03 065E6 00000000 (02695)
A04 065E8 00000000 (02696)
A05 065EA 00000000 (02697)
A06 065EC 00000000 (02698)
A07 065EE 00000000 (02699)
A08 065F0 00000000 (02700)
A09 065F2 00000000 (02701)
A10 065F4 00000000 (02702)
A11 065F6 00000000 (02703)
A12 065F8 00000000 (02704)
A13 065FA 00000000 (02705)
A14 065FC 00000000 (02706)
A15 065FE 00000000 (02707)
A16 06600 00000000 (02708)
A17 06602 00000000 (02709)
A18 06604 00000000 (02710)
A19 06606 00000000 (02711)
A20 06608 00000000 (02712)
A21 0660A 00000000 (02713)
A22 0660C 00000000 (02714)
A23 0660E 00000000 (02715)
A24 06610 00000000 (02716)
A25 06612 00000000 (02717)
A26 06614 00000000 (02718)
A27 06616 00000000 (02719)
A28 06618 00000000 (02720)
A29 0661A 00000000 (02721)
A30 0661C 00000000 (02722)
A31 0661E 00000000 (02723)
A32 06620 00000000 (02724)
A33 06622 00000000 (02725)
A34 06624 00000000 (02726)
A35 06626 00000000 (02727)
A36 06628 00000000 (02728)
A37 0662A 00000000 (02729)
A38 0662C 00000000 (02730)
A39 0662E 00000000 (02731)
A40 06630 00000000 (02732)
A41 06632 00000000 (02733)
A42 06634 00000000 (02734)
A43 06636 00000000 (02735)
A44 06638 00000000 (02736)
A45 0663A 00000000 (02737)
A46 0663C 00000000 (02738)
A47 0663E 00000000 (02739)
A48 06640 00000000 (02740)
A49 06642 00000000 (02741)
A50 06644 00000000 (02742)
A51 06646 00000000 (02743)
A52 06648 00000000 (02744)
A53 0664A 00000000 (02745)
A54 0664C 00000000 (02746)
A55 0664E 00000000 (02747)
A56 06650 00000000 (02748)
A57 06652 00000000 (02749)
A58 06654 00000000 (02750)
A59 06656 00000000 (02751)
A60 06658 00000000 (02752)
A61 0665A 00000000 (02753)
A62 0665C 00000000 (02754)
A63 0665E 00000000 (02755)
A64 06660 00000000 (02756)
A65 06662 00000000 (02757)
A66 06664 00000000 (02758)
A67 06666 00000000 (02759)
A68 06668 00000000 (02760)
A69 0666A 00000000 (02761)
A70 0666C 00000000 (02762)
A71 0666E 00000000 (02763)
A72 06670 00000000 (02764)
A73 06672 00000000 (02765)
A74 06674 00000000 (02766)
A75 06676 00000000 (02767)
A76 06678 00000000 (02768)
A77 0667A 00000000 (02769)
A78 0667C 00000000 (02770)
A79 0667E 00000000 (02771)
A80 06680 00000000 (02772)
A81 06682 00000000 (02773)
A82 06684 00000000 (02774)
A83 06686 00000000 (02775)
A84 06688 00000000 (02776)
A85 0668A 00000000 (02777)
A86 0668C 00000000 (02778)
A87 0668E 00000000 (02779)
A88 06690 00000000 (02780)
A89 06692 00000000 (02781)
A90 06694 00000000 (02782)
A91 06696 00000000 (02783)
A92 06698 00000000 (02784)
A93 0669A 00000000 (02785)
A94 0669C 00000000 (02786)
A95 0669E 00000000 (02787)
A96 066A0 00000000 (02788)
A97 066A2 00000000 (02789)
A98 066A4 00000000 (02790)
A99 066A6 00000000 (02791)
A00 066A8 00000000 (02792)
A01 066AA 00000000 (02793)
A02 066AC 00000000 (02794)
A03 066AE 00000000 (02795)
A04 066B0 00000000 (02796)
A05 066B2 00000000 (02797)
A06 066B4 00000000 (02798)
A07 066B6 00000000 (02799)
A08 066B8 00000000 (02800)
A09 066BA 00000000 (02801)
A10 066BC 00000000 (02802)
A11 066BE 00000000 (02803)
A12 066C0 00000000 (02804)
A13 066C2 00000000 (02805)
A14 066C4 00000000 (02806)
A15 066C6 00000000 (02807)
A16 066C8 00000000 (02808)
A17 066CA 00000000 (02809)
A18 066CC 00000000 (02810)
A19 066CE 00000000 (02811)
A20 066D0 00000000 (02812)
A21 066D2 00000000 (02813)
A22 066D4 00000000 (02814)
A23 066D6 00000000 (02815)
A24 066D8 00000000 (02816)
A25 066DA 00000000 (02817)
A26 066DC 00000000 (02818)
A27 066DE 00000000 (02819)
A28 066E0 00000000 (02820)
A29 066E2 00000000 (02821)
A30 066E4 00000000 (02822)
A31 066E6 00000000 (02823)
A32 066E8 00000000 (02824)
A33 066EA 00000000 (02825)
A34 066EC 00000000 (02826)
A35 066EE 00000000 (02827)
A36 066F0 00000000 (02828)
A37 066F2 00000000 (02829)
A38 066F4 00000000 (02830)
A39 066F6 00000000 (02831)
A40 066F8 00000000 (02832)
A41 066FA 00000000 (02833)
A42 066FC 00000000 (02834)
A43 066FE 00000000 (02835)
A44 06700 00000000 (02836)
A45 06702 00000000 (02837)
A46 06704 00000000 (02838)
A47 06706 00000000 (02839)
A48 06708 00000000 (02840)
A49 0670A 00000000 (02841)
A50 0670C 00000000 (02842)
A51 0670E 00000000 (02843)
A52 06710 00000000 (02844)
A53 06712 00000000 (02845)
A54 06714 00000000 (02846)
A55 06716 00000000 (02847)
A56 06718 00000000 (02848)
A57 0671A 00000000 (02849)
A58 0671C 00000000 (02850)
A59 0671E 00000000 (02851)
A60 06720 00000000 (02852)
A61 06722 00000000 (02853)
A62 06724 00000000 (02854)
A63 06726 00000000 (02855)
A64 06728 00000000 (02856)
A65 0672A 00000000 (02857)
A66 0672C 00000000 (02858)
A67 0672E 00000000 (02859)
A68 06730 00000000 (02860)
A69 06732 00000000 (02861)
A70 06734 00000000 (02862)
A71 06736 00000000 (02863)
A72 06738 00000000 (02864)
A73 0673A 00000000 (02865)
A74 0673C 00000000 (02866)
A75 0673E 00000000 (02867)
A76 06740 00000000 (02868)
A77 06742 00000000 (02869)
A78 06744 00000000 (02870)
A79 06746 00000000 (02871)
A80 06748 00000000 (02872)
A81 0674A 00000000 (02873)
A82 0674C 00000000 (02874)
A83 0674E 00000000 (02875)
A84 06750 00000000 (02876)
A85 06752 00000000 (02877)
A86 06754 00000000 (02878)
A87 06756 00000000 (02879)
A88 06758 00000000 (02880)
A89 0675A 00000000 (02881)
A90 0675C 00000000 (02882)
A91 0675E 00000000 (02883)
A92 06760 00000000 (02884)
A93 06762 00000000 (02885)
A94 06764 00000000 (02886)
A95 06766 00000000 (02887)
A96 06768 00000000 (02888)
A97 0676A 00000000 (02889)
A98 0676C 00000000 (02890)
A99 0676E 00000000 (02891)
A00 06770 00000000 (02892)
A01 06772 00000000 (02893)
A02 06774 00000000 (02894)
A03 06776 00000000 (02895)
A04 06778 00000000 (02896)
A05 0677A 00000000 (02897)
A06 0677C 00000000 (02898)
A07 0677E 00000000 (02899)
A08 06780 00000000 (02900)
A09 06782 00000000 (02901)
A10 06784 00000000 (02902)
A11 06786 00000000 (02903)
A12 06788 00000000 (02904)
A13 0678A 00000000 (02905)
A14 0678C 00000000 (02906)
A15 0678E 00000000 (02907)
A16 06790 00000000 (02908)
A17 06792 00000000 (02909)
A18 06794 00000000 (02910)
A19 06796 00000000 (02911)
A20 06798 00000000 (02912)
A21 0679A 00000000 (02913)
A22 0679C 00000000 (02914)
A23 0679E 00000000 (02915)
A24 067A0 00000000 (02916)
A25 067A2 00000000 (02917)
A26 067A4 00000000 (02918)
A27 067A6 00000000 (02919)
A28 067A8 00000000 (02920)
A29 067AA 00000000 (02921)
A30 067AC 00000000 (02922)
A31 067AE 00000000 (02923)
A32 067B0 00000000 (02924)
A33 067B2 00000000 (02925)
A34 067B4 00000000 (02926)
A35 067B6 00000000 (02927)
A36 067B8 00000000 (02928)
A37 067BA 00000000 (02929)
A38 067BC 00000000 (02930)
A39 067BE 00000000 (02931)
A40 067C0 00000000 (02932)
A41 067C2 00000000 (02933)
A42 067C4 00000000 (02934)
A43 067C6 00000000 (02935)
A44 067C8 00000000 (02936)
A45 067CA 00000000 (02937)
A46 067CC 00000000 (02938)
A47 067CE 00000000 (02939)
A48 067D0 00000000 (02940)
A49 067D2 00000000 (02941)
A50 067D4 00000000 (02942)
A51 067D6 00000000 (02943)
A52 067D8 00000000 (02944)
A53 067DA 00000000 (02945)
A54 067DC 00000000 (02946)
A55 067DE 00000000 (02947)
A56 067E0 00000000 (02948)
A57 067E2 00000000 (02949)
A58 067E4 00000000 (02950)
A59 067E6 00000000 (02951)
A60 067E8 00000000 (02952)
A61 067EA 00000000 (02953)
A62 067EC 00000000 (02954)
A63 067EE 00000000 (02955)
A64 067F0 00000000 (02956)
A65 067F2 00000000 (02957)
A66 067F4 00000000 (02958)
A67 067F6 00000000 (02959)
A68 067F8 00000000 (02960)
A69 067FA 00000000 (02961)
A70 067FC 00000000 (02962)
A71 067FE 00000000 (02963)
A72 06800 00000000 (02964)
A73 06802 00000000 (02965)
A74 06804 00000000 (02966)
A75 06806 00000000 (02967)
A76 06808 00000000 (02968)
A77 0680A 00000000 (02969)
A78 0680C 00000000 (02970)
A79 0680E 00000000 (02971)
A80 06810 00000000 (02972)
A81 06812 00000000 (02973)
A82 06814 00000000 (02974)
A83 06816 00000000 (02975)
A84 06818 00000000 (02976)
A85 0681A 00000000 (02977)
A86 0681C 00000000 (02978)
A87 0681E 00000000 (02979)
A88 06820 00000000 (02980)
A89 06822 00000000 (02981)
A90 06824 00000000 (02982)
A91 06826 00000000 (02983)
A92 06828 00000000 (02984)
A93 0682A 00000000 (02985)
A94 0682C 00000000 (02986)
A95 0682E 00000000 (02987)
A96 06830 00000000 (02988)
A97 06832 00000000 (02989)
A98 06834 00000000 (02990)
A99 06836 00000000 (02991)
A00 06838 00000000 (02992)
A01 0683A 00000000 (02993)
A02 0683C 00000000 (02994)
A03 0683E 00000000 (02995)
A04 06840 00000000 (02996)
A05 06842 00000000 (02997)
A06 06844 00000000 (02998)
A07 06846 00000000 (02999)
A08 06848 00000000 (03000)
A09 0684A 00000000 (03001)
A10 0684C 00000000 (03002)
A11 0684E 00000000 (03003)
A12 06850 00000000 (03004)
A13 06852 00000000 (03005)
A14 06854 00000000 (03006)
A15 06856 00000000 (03007)
A16 06858 00000000 (03008)
A17 0685A 00000000 (03009)
A18 0685C 00000000 (03010)
A19 0685E 00000000 (03011)
A20 06860 00000000 (03012)
A21 06862 00000000 (03013)
A22 06864 00000000 (03014)
A23 06866 00000000 (03015)
A24 06868 00000000 (03016)
A25 0686A 00000000 (03017)
A26 0686C 00000000 (03018)
A27 0686E 00000000 (03019)
A28 06870 00000000 (03020)
A29 06872 00000000 (03021)
A30 06874 00000000 (03022)
A31 06876 00000000 (03023)
A32 06878 00000000 (03024)
A33 0687A 00000000 (03025)
A34 0687C 00000000 (03026)
A35 0687E 00000000 (03027)
A36 06880 00000000 (03028)
A37 06882 00000000 (03029)
A38 06884 00000000 (03030)
A39 06886 00000000 (03031)
A40 06888 00000000 (03032)
A41 0688A 00000000 (03033)
A42 0688C 00000000 (03034)
A43 0688E 00000000 (03035)
A44 06890 00000000 (03036)
A45 06892 00000000 (03037)
A46 06894 00000000 (03038)
A47 06896 00000000 (03039)
A48 06898 00000000 (03040)
A49 0689A 00000000 (03041)
A50 0689C 00000000 (03042)
A51 0689E 00000000 (03043)
A52 068A0 00000000 (03044)
A53 068A2 00000000 (03045)
A54 068A4 00000000 (03046)
A55 068A6 00000000 (03047)
A56 068A8 00000000 (03048)
A57 068AA 00000000 (03049)
A58 068AC 00000000 (03050)
A59 068AE 00000000 (03051)
A60 068B0 00000000 (03052)
A61 068B2 00
```



```

PAGE 611 (00ND)(<DC116>00NIGN.MSO.2, 29-Dec-88 14:30:40, E4: WOLF
APSJ-INVP(Y,A,B,V) APS PROGRAM FOR INVP
A13 06500 26090025 (02603) SUB0(BR0,BR2) POINT TO V(B-N)
A14 0650A 2050100E (02604) SUBA
A15 0650C 2A700000 (02605) (UNUSED) VBS-1
A16 0650E 2C120000 (02606) SUB(BR1,MSS)
A17 06590 2F1A0100 (02607) ADD(BR1,216*W$)
A18 06592 30700000 (02608) LOAD(BR2,C0)
A19 06594 37000000 (02609) LOAD(BR2,MSS)
A20 06596 34700000 (02691) MAIN LOOP
A21 06598 3609003A (02692) B4: ADDL(BR0,W$,TF)
A22 0659A 3099003A (02693) ADDL(BR1,W$,TF)
A23 0659C 3A99003A (02694) ADDL(BR1,W$,TF)
A24 0659E 3C090032 (02695) ADDL(BR1,W$,TF)
A25 065A0 3F090032 (02696) ADDL(BR1,W$,TF)
A26 065A2 4009003E (02697) ADDL(BR0,3-W$,TF)
A27 065A4 47291002 (02698) SUBL(BR2,2),JUMPP(B4)
A28 065A6 44200031 (02699) CLEAR(R1)
A29 065A8 46000020 (02700) NOP(0)
A30 065AA 40300032 (02701) )
A31 065AC 4B726505 (02702) )
A32 065AE 4CF3EFC0 (02703) JUMPTUT PROGRAM
A33 065B0 4FF20794 (02704) ) BWS: V(W)
A34 065B2 50F20794 (02705) ) BWS: LOOP COUNTER (STARTS AT VBS-1)
A35 065B4 52400022 (02706) ) BWS: SCRATCH
A36 065B6 54500000 (02707) )
A37 065B8 56020000 (02708) INVP50: SET(RA)
A38 065BA 580A0006 (02709) JEMPERATE BUS1 ADR FOR THE INSTRUCTION TO BE MODIFIED (RIGHT HALF)
A39 065BC 5A112C01 (02710) LOAD(BW3,INVP50*W$(1),TE)
A40 065BE 5C200030 (02711) JEMPERATE PSEUDOMEMORY ADR FOR LOADING THE APS MEMORY
A41 065C0 5F000020 (02712) LOAD(BW3,APSWEN(1),TF)
A42 065C2 6000450A (02713) NOW GET ON WITH THE "ORDINARY" OUTPUT
A43 065C4 60000000 (02714) LOAD(BW3,OW$(1),TF)
A44 065C6 60000000 (02715) LOAD(BW3,OW$(1),TF)
A45 065C8 60000000 (02716) LOAD(BW3,OW$(1),TF)
A46 065CA 60000000 (02717) LOAD(BW3,OW$(1),TF)
A47 065CC 60000000 (02718) LOAD(BW3,OW$(1),TF)
A48 065CE 60000000 (02719) SUB(BW0,MSS)
A49 065D0 60000000 (02720) JUMPTUT LOOP
A50 065D2 60000000 (02721) B5: ADD(BW0,C0),TF)
A51 065D4 60000000 (02722) SUBL(BW1,1),JUMPP(B5)
A52 065D6 60000000 (02723) CLFAR(RO)
A53 065D8 60000000 (02724) NOP(0)
A54 065DA 60000000 (02725) END
A55 065DC 60000000 (02726) INVP50: DATA 7F*0.0"
A56 065DE 60000000 (02727) INVP52: 0L-INVP51
A57 065DF 60000000 (02728) INVP53: 0L-INVP51
A58 065E0 60000000 (02729) INVP54: 0L-INVP51
A59 065E2 60000000 (02730) INVP55: 0L-INVP51
A60 065E4 60000000 (02731) INVP56: 0L-INVP51
A61 065E6 60000000 (02732) INVP57: 0L-INVP51
A62 065E8 60000000 (02733) INVP58: 0L-INVP51
A63 065EA 60000000 (02734) INVP59: 0L-INVP51
A64 065EC 60000000 (02735) INVP60: 0L-INVP51
A65 065EE 60000000 (02736) INVP61: 0L-INVP51
A66 065F0 60000000 (02737) INVP62: 0L-INVP51
A67 065F2 60000000 (02738) INVP63: 0L-INVP51
A68 065F4 60000000 (02739) INVP64: 0L-INVP51
A69 065F6 60000000 (02740) INVP65: 0L-INVP51
A70 065F8 60000000 (02741) INVP66: 0L-INVP51
A71 065FA 60000000 (02742) INVP67: 0L-INVP51
A72 065FC 60000000 (02743) INVP68: 0L-INVP51
A73 065FE 60000000 (02744) INVP69: 0L-INVP51
A74 06600 60000000 (02745) INVP70: 0L-INVP51
A75 06602 60000000 (02746) INVP71: 0L-INVP51
A76 06604 60000000 (02747) INVP72: 0L-INVP51
A77 06606 60000000 (02748) INVP73: 0L-INVP51
A78 06608 60000000 (02749) INVP74: 0L-INVP51
A79 0660A 60000000 (02750) INVP75: 0L-INVP51
A80 0660C 60000000 (02751) INVP76: 0L-INVP51
A81 0660E 60000000 (02752) INVP77: 0L-INVP51
A82 06610 60000000 (02753) INVP78: 0L-INVP51
A83 06612 60000000 (02754) INVP79: 0L-INVP51
A84 06614 60000000 (02755) INVP80: 0L-INVP51
A85 06616 60000000 (02756) INVP81: 0L-INVP51
A86 06618 60000000 (02757) INVP82: 0L-INVP51
A87 0661A 60000000 (02758) INVP83: 0L-INVP51
A88 0661C 60000000 (02759) INVP84: 0L-INVP51
A89 0661E 60000000 (02760) INVP85: 0L-INVP51
A90 06620 60000000 (02761) INVP86: 0L-INVP51
A91 06622 60000000 (02762) INVP87: 0L-INVP51
A92 06624 60000000 (02763) INVP88: 0L-INVP51
A93 06626 60000000 (02764) INVP89: 0L-INVP51
A94 06628 60000000 (02765) INVP90: 0L-INVP51
A95 0662A 60000000 (02766) INVP91: 0L-INVP51
A96 0662C 60000000 (02767) INVP92: 0L-INVP51
A97 0662E 60000000 (02768) INVP93: 0L-INVP51
A98 06630 60000000 (02769) INVP94: 0L-INVP51
A99 06632 60000000 (02770) INVP95: 0L-INVP51
A100 06634 60000000 (02771) INVP96: 0L-INVP51
A101 06636 60000000 (02772) INVP97: 0L-INVP51
A102 06638 60000000 (02773) INVP98: 0L-INVP51
A103 0663A 60000000 (02774) INVP99: 0L-INVP51
A104 0663C 60000000 (02775) INVP100: 0L-INVP51
A105 0663E 60000000 (02776) INVP101: 0L-INVP51
A106 06640 60000000 (02777) INVP102: 0L-INVP51
A107 06642 60000000 (02778) INVP103: 0L-INVP51
A108 06644 60000000 (02779) INVP104: 0L-INVP51
A109 06646 60000000 (02780) INVP105: 0L-INVP51
A110 06648 60000000 (02781) INVP106: 0L-INVP51
A111 0664A 60000000 (02782) INVP107: 0L-INVP51
A112 0664C 60000000 (02783) INVP108: 0L-INVP51
A113 0664E 60000000 (02784) INVP109: 0L-INVP51
A114 06650 60000000 (02785) INVP110: 0L-INVP51
A115 06652 60000000 (02786) INVP111: 0L-INVP51
A116 06654 60000000 (02787) INVP112: 0L-INVP51
A117 06656 60000000 (02788) INVP113: 0L-INVP51
A118 06658 60000000 (02789) INVP114: 0L-INVP51
A119 0665A 60000000 (02790) INVP115: 0L-INVP51
A120 0665C 60000000 (02791) INVP116: 0L-INVP51
A121 0665E 60000000 (02792) INVP117: 0L-INVP51
A122 06660 60000000 (02793) INVP118: 0L-INVP51
A123 06662 60000000 (02794) INVP119: 0L-INVP51
A124 06664 60000000 (02795) INVP120: 0L-INVP51
A125 06666 60000000 (02796) INVP121: 0L-INVP51
A126 06668 60000000 (02797) INVP122: 0L-INVP51
A127 0666A 60000000 (02798) INVP123: 0L-INVP51
A128 0666C 60000000 (02799) INVP124: 0L-INVP51
A129 0666E 60000000 (02800) INVP125: 0L-INVP51
A130 06670 60000000 (02801) INVP126: 0L-INVP51
A131 06672 60000000 (02802) INVP127: 0L-INVP51
A132 06674 60000000 (02803) INVP128: 0L-INVP51
A133 06676 60000000 (02804) INVP129: 0L-INVP51
A134 06678 60000000 (02805) INVP130: 0L-INVP51
A135 0667A 60000000 (02806) INVP131: 0L-INVP51
A136 0667C 60000000 (02807) INVP132: 0L-INVP51
A137 0667E 60000000 (02808) INVP133: 0L-INVP51
A138 06680 60000000 (02809) INVP134: 0L-INVP51
A139 06682 60000000 (02810) INVP135: 0L-INVP51
A140 06684 60000000 (02811) INVP136: 0L-INVP51
A141 06686 60000000 (02812) INVP137: 0L-INVP51
A142 06688 60000000 (02813) INVP138: 0L-INVP51
A143 0668A 60000000 (02814) INVP139: 0L-INVP51
A144 0668C 60000000 (02815) INVP140: 0L-INVP51
A145 0668E 60000000 (02816) INVP141: 0L-INVP51
A146 06690 60000000 (02817) INVP142: 0L-INVP51
A147 06692 60000000 (02818) INVP143: 0L-INVP51
A148 06694 60000000 (02819) INVP144: 0L-INVP51
A149 06696 60000000 (02820) INVP145: 0L-INVP51
A150 06698 60000000 (02821) INVP146: 0L-INVP51
A151 0669A 60000000 (02822) INVP147: 0L-INVP51
A152 0669C 60000000 (02823) INVP148: 0L-INVP51
A153 0669E 60000000 (02824) INVP149: 0L-INVP51
A154 066A0 60000000 (02825) INVP150: 0L-INVP51
A155 066A2 60000000 (02826) INVP151: 0L-INVP51
A156 066A4 60000000 (02827) INVP152: 0L-INVP51
A157 066A6 60000000 (02828) INVP153: 0L-INVP51
A158 066A8 60000000 (02829) INVP154: 0L-INVP51
A159 066AA 60000000 (02830) INVP155: 0L-INVP51
A160 066AC 60000000 (02831) INVP156: 0L-INVP51
A161 066AE 60000000 (02832) INVP157: 0L-INVP51
A162 066B0 60000000 (02833) INVP158: 0L-INVP51
A163 066B2 60000000 (02834) INVP159: 0L-INVP51
A164 066B4 60000000 (02835) INVP160: 0L-INVP51
A165 066B6 60000000 (02836) INVP161: 0L-INVP51
A166 066B8 60000000 (02837) INVP162: 0L-INVP51
A167 066BA 60000000 (02838) INVP163: 0L-INVP51
A168 066BC 60000000 (02839) INVP164: 0L-INVP51
A169 066BE 60000000 (02840) INVP165: 0L-INVP51
A170 066C0 60000000 (02841) INVP166: 0L-INVP51
A171 066C2 60000000 (02842) INVP167: 0L-INVP51
A172 066C4 60000000 (02843) INVP168: 0L-INVP51
A173 066C6 60000000 (02844) INVP169: 0L-INVP51
A174 066C8 60000000 (02845) INVP170: 0L-INVP51
A175 066CA 60000000 (02846) INVP171: 0L-INVP51
A176 066CC 60000000 (02847) INVP172: 0L-INVP51
A177 066CE 60000000 (02848) INVP173: 0L-INVP51
A178 066D0 60000000 (02849) INVP174: 0L-INVP51
A179 066D2 60000000 (02850) INVP175: 0L-INVP51
A180 066D4 60000000 (02851) INVP176: 0L-INVP51
A181 066D6 60000000 (02852) INVP177: 0L-INVP51
A182 066D8 60000000 (02853) INVP178: 0L-INVP51
A183 066DA 60000000 (02854) INVP179: 0L-INVP51
A184 066DC 60000000 (02855) INVP180: 0L-INVP51
A185 066DE 60000000 (02856) INVP181: 0L-INVP51
A186 066E0 60000000 (02857) INVP182: 0L-INVP51
A187 066E2 60000000 (02858) INVP183: 0L-INVP51
A188 066E4 60000000 (02859) INVP184: 0L-INVP51
A189 066E6 60000000 (02860) INVP185: 0L-INVP51
A190 066E8 60000000 (02861) INVP186: 0L-INVP51
A191 066EA 60000000 (02862) INVP187: 0L-INVP51
A192 066EC 60000000 (02863) INVP188: 0L-INVP51
A193 066EE 60000000 (02864) INVP189: 0L-INVP51
A194 066F0 60000000 (02865) INVP190: 0L-INVP51
A195 066F2 60000000 (02866) INVP191: 0L-INVP51
A196 066F4 60000000 (02867) INVP192: 0L-INVP51
A197 066F6 60000000 (02868) INVP193: 0L-INVP51
A198 066F8 60000000 (02869) INVP194: 0L-INVP51
A199 066FA 60000000 (02870) INVP195: 0L-INVP51
A200 066FC 60000000 (02871) INVP196: 0L-INVP51
A201 066FE 60000000 (02872) INVP197: 0L-INVP51
A202 06700 60000000 (02873) INVP198: 0L-INVP51
A203 06702 60000000 (02874) INVP199: 0L-INVP51
A204 06704 60000000 (02875) INVP200: 0L-INVP51
A205 06706 60000000 (02876) INVP201: 0L-INVP51
A206 06708 60000000 (02877) INVP202: 0L-INVP51
A207 0670A 60000000 (02878) INVP203: 0L-INVP51
A208 0670C 60000000 (02879) INVP204: 0L-INVP51
A209 0670E 60000000 (02880) INVP205: 0L-INVP51
A210 06710 60000000 (02881) INVP206: 0L-INVP51
A211 06712 60000000 (02882) INVP207: 0L-INVP51
A212 06714 60000000 (02883) INVP208: 0L-INVP51
A213 06716 60000000 (02884) INVP209: 0L-INVP51
A214 06718 60000000 (02885) INVP210: 0L-INVP51
A215 0671A 60000000 (02886) INVP211: 0L-INVP51
A216 0671C 60000000 (02887) INVP212: 0L-INVP51
A217 0671E 60000000 (02888) INVP213: 0L-INVP51
A218 06720 60000000 (02889) INVP214: 0L-INVP51
A219 06722 60000000 (02890) INVP215: 0L-INVP51
A220 06724 60000000 (02891) INVP216: 0L-INVP51
A221 06726 60000000 (02892) INVP217: 0L-INVP51
A222 06728 60000000 (02893) INVP218: 0L-INVP51
A223 0672A 60000000 (02894) INVP219: 0L-INVP51
A224 0672C 60000000 (02895) INVP220: 0L-INVP51
A225 0672E 60000000 (02896) INVP221: 0L-INVP51
A226 06730 60000000 (02897) INVP222: 0L-INVP51
A227 06732 60000000 (02898) INVP223: 0L-INVP51
A228 06734 60000000 (02899) INVP224: 0L-INVP51
A229 06736 60000000 (02900) INVP225: 0L-INVP51
A230 06738 60000000 (02901) INVP226: 0L-INVP51
A231 0673A 60000000 (02902) INVP227: 0L-INVP51
A232 0673C 60000000 (02903) INVP228: 0L-INVP51
A233 0673E 60000000 (02904) INVP229: 0L-INVP51
A234 06740 60000000 (02905) INVP230: 0L-INVP51
A235 06742 60000000 (02906) INVP231: 0L-INVP51
A236 06744 60000000 (02907) INVP232: 0L-INVP51
A237 06746 60000000 (02908) INVP233: 0L-INVP51
A238 06748 60000000 (02909) INVP234: 0L-INVP51
A239 0674A 60000000 (02910) INVP235: 0L-INVP51
A240 0674C 60000000 (02911) INVP236: 0L-INVP51
A241 0674E 60000000 (02912) INVP237: 0L-INVP51
A242 06750 60000000 (02913) INVP238: 0L-INVP51
A243 06752 60000000 (02914) INVP239: 0L-INVP51
A244 06754 60000000 (02915) INVP240: 0L-INVP51
A245 06756 60000000 (02916) INVP241: 0L-INVP51
A246 06758 60000000 (02917) INVP242: 0L-INVP51
A247 0675A 60000000 (02918) INVP243: 0L-INVP51
A248 0675C 60000000 (02919) INVP244: 0L-INVP51
A249 0675E 60000000 (02920) INVP245: 0L-INVP51
A250 06760 60000000 (02921) INVP246: 0L-INVP51
A251 06762 60000000 (02922) INVP247: 0L-INVP51
A252 06764 60000000 (02923) INVP248: 0L-INVP51
A253 06766 60000000 (02924) INVP249: 0L-INVP51
A254 06768 60000000 (02925) INVP250: 0L-INVP51
A255 0676A 60000000 (02926) INVP251: 0L-INVP51
A256 0676C 60000000 (02927) INVP252: 0L-INVP51
A257 0676E 60000000 (02928) INVP253: 0L-INVP51
A258 06770 60000000 (02929) INVP254: 0L-INVP51
A259 06772 60000000 (02930) INVP255: 0L-INVP51
A260 06774 60000000 (02931) INVP256: 0L-INVP51
A261 06776 60000000 (02932) INVP257: 0L-INVP51
A262 06778 60000000 (02933) INVP258: 0L-INVP51
A263 0677A 60000000 (02934) INVP259: 0L-INVP51
A264 0677C 60000000 (02935) INVP260: 0L-INVP51
A265 0677E 60000000 (02936) INVP261: 0L-INVP51
A266 06780 60000000 (02937) INVP262: 0L-INVP51
A267 06782 60000000 (02938) INVP263: 0L-INVP51
A268 06784 60000000 (02939) INVP264: 0L-INVP51
A269 06786 60000000 (02940) INVP265: 0L-INVP51
A270 06788 60000000 (02941) INVP266: 0L-INVP51
A271 0678A 60000000 (02942) INVP267: 0L-INVP51
A272 0678C 60000000 (02943) INVP268: 0L-INVP51
A273 0678E 60000000 (02944) INVP269: 0L-INVP51
A274 06790 60000000 (02945) INVP270: 0L-INVP51
A275 06792 60000000 (02946) INVP271: 0L-INVP51
A276 06794 60000000 (02947) INVP272: 0L-INVP51
A277 06796 60000000 (02948) INVP273: 0L-INVP51
A278 06798 60000000 (02949) INVP274: 0L-INVP51
A279 0679A 60000000 (02950) INVP275: 0L-INVP51
A280 0679C 60000000 (02951) INVP276: 0L-INVP51
A281 0679E 60000000 (02952) INVP277: 0L-INVP51
A282 067A0 60000000 (02953) INVP278: 0L-INVP51
A283 067A2 60000000 (02954) INVP279: 0L-INVP51
A284 067A4 60000000 (02955) INVP280: 0L-INVP51
A285 067A6 60000000 (02956) INVP281: 0L-INVP51
A286 067A8 60000000 (02957) INVP282: 0L-INVP51
A287 067AA 60000000 (02958) INVP283: 0L-INVP51
A288 067AC 60000000 (02959) INVP284: 0L-INVP51
A289 067AE 60000000 (02960) INVP285: 0L-INVP51
A290 067B0 60000000 (02961) INVP286: 0L-INVP51
A291 067B2 60000000 (02962) INVP287: 0L-INVP51
A292 067B4 60000000 (02963) INVP288: 0L-INVP51
A293 067B6 60000000 (02964) INVP289: 0L-INVP51
A294 067B8 60000000 (02965) INVP290: 0L-INVP51
A295 067BA 60000000 (02966) INVP291: 0L-INVP51
A296 067BC 60000000 (02967) INVP292: 0L-INVP51
A297 067BE 60000000 (02968) INVP293: 0L-INVP51
A298 067C0 60000000 (02969) INVP294: 0L-INVP51
A299 067C2 60000000 (02970) INVP295: 0L-INVP51
A300 067C4 60000000 (02971) INVP296: 0L-INVP51
A301 067C6 60000000 (02972) INVP297: 0L-INVP51
A302 067C8 60000000 (02973) INVP298: 0L-INVP51
A303 067CA 60000000 (02974) INVP299: 0L-INVP51
A304 067CC 60000000 (02975) INVP300: 0L-INVP51
A305 067CE 60000000 (02976) INVP301: 0L-INVP51
A306 067D0 60000000 (02977) INVP302: 0L-INVP51
A307 067D2 60000000 (02978) INVP303: 0L-INVP51
A308 067D4 60000000 (02979) INVP304: 0L-INVP51
A309 067D6 60000000 (02980) INVP305: 0L-INVP51
A310 067D8 60000000 (02981) INVP306: 0L-INVP51
A311 067DA 60000000 (02982) INVP307: 0L-INVP51
A312 067DC 60000000 (02983) INVP308: 0L-INVP51
A313 067DE 60000000 (02984) INVP309: 0L-INVP51
A314 067E0 60000000 (02985) INVP310: 0L-INVP51
A315 067E2 60000000 (02986) INVP311: 0L-INVP51
A316 067E4 60000000 (02987) INVP312: 0L-INVP51
A317 067E6 60000000 (02988) INVP313: 0L-INVP51
A318 067E8 60000000 (02989) INVP314: 0L-INVP51
A319 067EA 60000000 (02990) INVP315: 0L-INVP51
A320 067EC 60000000 (02991) INVP316: 0L-INVP51
A321 067EE 60000000 (02992) INVP317: 0L-INVP51
A322 067F0 60000000 (02993) INVP318: 0L-INVP51
A323 067F2 60000000 (02994) INVP319: 0L-INVP51
A324 067F4 60000000 (02995) INVP320: 0L-INVP51
A325 067F6 60000000 (02996) INVP321: 0L-INVP51
A326 067F8 60000000 (02997) INVP322: 0L-INVP51
A327 067FA 60000000 (02998) INVP323: 0L-INVP51
A328 067FC 60000000 (02999) INVP324: 0L-INVP51
A329 067FE 60000000
```


(02720) - AP3-ADPEAK(U) MONITOR PEAK ABS VAL OF INPUT SPEECH
(02729) ; J WOLF, 10/30/80

1027307 ; BINDS TO APS3-ADPEAK

PERFORMS A RUNNING MAXIMUM-ABSOLUTE-VALUE SCAN OF A BUFFER OF SPEECH DATA INPUT FROM THE ADAM, TO PERMIT MONITORING OF THE (PSR) LEVEL OF THE INPUT SIGNAL. THE RUNNING PEAK IS MAINTAINED AS AN INTEGER (VALUE = 0-2047), FOR EASE OF DISPLAY TO THE USER. THE A/D SIGNAL VALUES RANGE FROM -1 TO +1, SO THERE MUST BE BOTH A MAGNITUDE CHANGE (OF 2ⁿ-1) AND AN INTEGER/FLTPT CHANGE WHEN THIS RUNNING MAX IS BROUGHT IN AND WRITTEN OUT AGAIN.

```

      ... , INPUT SEQUENCE:
      (02741) , PREVIOUS MAX (INTEGER), U(0), ..., U(UBS-1), CFI)
      (02742) , PREVIOUS MAX (INTEGER), U(0), ..., U(UBS-1), CFI)

```

```

(02744) , NEW MAX (INTEGER), [ED]

```

2 ELEMENTS PER 7 APU CYCLES = -.35 USEC/ELEMENT
1 MEM REF PER ELEMENT = .5 USEC (DMS 1) PER ELEMENT

(02749) ; LEFT
RIGHT

002730	11:00	IMAX(000)	IMAX(EVEN)
002751	11:10	IMAX(000)	IMAX(EVEN)

(02753) ; A2: IMAXI (EVEN)
(02753) ; A2: IMAXI (EVEN)

MAXI (OVERALL)	MAXI / 16
102754	143
102755	144

NAME
(2560)
(95120)

(02758) 1141 108KUS\$A
 (02759) 1141 108KUS\$A

(02760) ;
03761 ;

```

(02762) ADPKU$A: MOV(IQA,A1)

```

NOV(A1), MORH(A1)

```

(02766) #1:      NOP \ MOV(1QA,1B)

```

(02768) JUMPS(EXH10,FI)

(02770) MAXABS(A0,A1) \ 1

```

(02772) EXMID: MOV(R,A1) \ MOV(

```

(02774) MIN(A1,A2) \ MIN
 (02775) MAX(A1,A2) \ MAX

```

(021131)
MOV(A3),DECN(23)
MOV(A4),ALIGN(A4)
(02776)

```

```

(02777)
(02778)
MOV(W,DQ) \ MUF
CLEAR(RA)

```

(0279) 10PKUS2=01-10PKUS1

JAPU	START	ADDRESS
JAPU	MODULE	SIZE
0000	0000	0000
0001	0001	0001
0002	0002	0002
0003	0003	0003
0004	0004	0004
0005	0005	0005
0006	0006	0006
0007	0007	0007
0008	0008	0008
0009	0009	0009
0010	0010	0010
0011	0011	0011
0012	0012	0012
0013	0013	0013
0014	0014	0014
0015	0015	0015
0016	0016	0016
0017	0017	0017
0018	0018	0018
0019	0019	0019
0020	0020	0020
0021	0021	0021
0022	0022	0022
0023	0023	0023
0024	0024	0024
0025	0025	0025
0026	0026	0026
0027	0027	0027
0028	0028	0028
0029	0029	0029
0030	0030	0030
0031	0031	0031
0032	0032	0032
0033	0033	0033
0034	0034	0034
0035	0035	0035
0036	0036	0036
0037	0037	0037
0038	0038	0038
0039	0039	0039
0040	0040	0040
0041	0041	0041
0042	0042	0042
0043	0043	0043
0044	0044	0044
0045	0045	0045
0046	0046	0046
0047	0047	0047
0048	0048	0048
0049	0049	0049
0050	0050	0050
0051	0051	0051
0052	0052	0052
0053	0053	0053
0054	0054	0054
0055	0055	0055
0056	0056	0056
0057	0057	0057
0058	0058	0058
0059	0059	0059
0060	0060	0060
0061	0061	0061
0062	0062	0062
0063	0063	0063
0064	0064	0064
0065	0065	0065
0066	0066	0066
0067	0067	0067
0068	0068	0068
0069	0069	0069
0070	0070	0070
0071	0071	0071
0072	0072	0072
0073	0073	0073
0074	0074	0074
0075	0075	0075
0076	0076	0076
0077	0077	0077
0078	0078	0078
0079	0079	0079
0080	0080	0080
0081	0081	0081
0082	0082	0082
0083	0083	0083
0084	0084	0084
0085	0085	0085
0086	0086	0086
0087	0087	0087
0088	0088	0088
0089	0089	0089
0090	0090	0090
0091	0091	0091
0092	0092	0092
0093	0093	0093
0094	0094	0094
0095	0095	0095
0096	0096	0096
0097	0097	0097
0098	0098	0098
0099	0099	0099

```

JFETCH PREVIOUS MAX (INTEGER)
IMULT BY (2**15)*(2**11)
JAND NORMALIZE FOR GOOD MEASURE
JII = PREVIOUS MAX (FLT PT)
J-- \ U(EVEN)
JSTORE MAX(ODD) \ MAKE NEW MAX(EVEN)
JLEAP IF ODD NO. OF ELEMENTS IN U
JFETCH U(ODD) \ --
JMAKE NEW MAX(ODD) \ STORE MAX(EVEN)
JLOOP IF MORE DATA IN U
JSTORE MAX(ODD) \ BRING OVER MAX(EVEN)
JSTORE MAX(EVEN)
JRESOLVE MAX(U(ODD),U(EVEN))
IMULT NEW MAX BY (2**15)*(2**11)
JAND CONVERT TO INTEGER
JAND OUTPUT IT
JAND HALT

```

DATA DATA

```

      JFETCH PREVIOUS MAX (INTEGER)
      JMULT BY (2--15)*(2--11)
      JAND NORMALIZE FOR GOOD MEASURE
      JAI = PREVIOUS MAX (FLT PT)
      J-- U(VEEN)
      JSTORE MAX(ODD) \ MAKE NEW MAX(EVEN)
      JUMP IF ODD NO. OF ELEMENTS IN V
      JFETCH U(ODD) \ --
      JMAKE NEW MAX(ODD) \ STORE MAX(EVEN)
      JLOOP IF MORE DATA IN V
      JSTORE MAX(ODD) \ BRING OVER MAX(EVEN)
      JSTORE MAX(EVEN)
      JRESOLVE MAX(U(ODD),U(EVEN))
      JMULT NEW MAX BY (2--15)*(2--11)
      JAND CONVERT TO INTEGER
      JAND OUTPUT IT
      JAND HALT
      JUMP(0)
      RA-ADPKUSSA

```

06500 0000
06501 0012

A00	05D2	00F100F1
A01	05D4	22D22220
A02	06D6	32333231
A03	05D8	00910091
A04	05DA	00900090
A05	05DC	00971100
A06	06DE	91100000
A07	05E0	00F00000
A08	05F2	71000091
A09	05E4	90F00000
A0A	05F6	00910090
A0B	05F8	00520000
A0C	05EA	62200000
A0D	05EC	2A730000
A0E	05EE	3A940000
A0F	05F0	009C0000
A10	05F2	20322032
A11	05F4	10000012
		00000012

94596 (02701)

PAGE 64: (END)(<0C116>BN16M.MSD.2, 29-Dec-88 14:38:40, Ed: WOLF
APS3-ADPEAK

```

(02782) APS3-ADPEAK
(02783) J. WOLF, 10/30/88
(02784) B1MDS TO APS3-ADPEAK
(02785)
(02786) JWE MAINTAIN THE RUNNING ABS. MAX. AS AN INTEGER IN THE INTEGER
(02787) SCALAR TABLE, FOR CONVENIENCE IN DISPLAYING ITS VALUE TO THE USER.
(02788) BECAUSE SNAP STANDARD BINDING DOESN'T ALLOW FOR INTEGER SCALARS
(02789) AS ARGUMENTS TO ARRAY FUNCTIONS, WE SIMPLY "HARD WIRE" THE
(02790) INTEGER SCALAR OFFSET OF TADPK TO THIS APS PROGRAM.
(02791) TADPK=87
00000057 (02792) TADPK=87
(02793)
(02794) JHEADER BLOCK
(02795)
(02796) EVEN
(02797) ADDR ADPKAS1
(02798) ADDR B
(02799) DATA B
(02800) DATA ADPKAS2
(02801) ADDR ADPKAS3
(02802) EVEN
(02803) JINPUT PROGRAM:
(02804) B00: B(0)
(02805) B01: LOOP COUNTER
(02806)
(02807) ADPKAS1
(02808) ADPKAS1: BEGIN APS(ADPKAS1)
(02809) JSM(ADPKAS1,P2)
(02810) SET(B0)
(02811) LOAD(B0B,ISUT$TADPK(1),TE) JGEN ADR FOR TADPK
(02812) R10: LOAD(B0B,C13) JLOAD B BASE ADR
(02813) LOAD(B0B,MS) JBASE-SPACING
(02814) SUB(B0B,MS) JGEN B(0)
(02815) R1: ADD(B0B,C9,TF) JLOOP TIL B0S ELEMENTS ARE DONE
(02816) SOBL(B0B,1),JUMPP(R1)
(02817) CLEAR(B1)
(02818) NOP(B)
(02819)
(02820) JOUTPUT PROGRAM: WRITE NEW MAX BACK IN THE INTEGER SCALAR TABLE
(02821)
(02822) ADPKAS1: SET(RA)
(02823) LOAD(B0B,ISUT$TADPK(1),TE)
(02824) CLEAR(B0)
(02825) NOP(B)
(02826)
(02827) ADPKAS1=BC
(02828) END
0661A 06600000 ADPKAS1: DATA 2F'0.0'
...
00000020 (02830) ADPKAS2 = BL - ADPKA

```

JPTR TO CONSTRUCTED INSTRUCTION BLOCK
JNO (REAL) SCALARS USED
JAPS MODULE SIZE
JPTR TO CHAIN ANCHOR

JSET AND START OUTPUT PROGRAM

JCHAIN ANCHOR

JSTORAGE FOR CONSTRUCTED INSTRUCTIONS

```

PAGE 65: (00ND)C0C116>R0116M.N50.2, 29-Dec-88 14:38:48, Ed: WOLF
          APDMP(V) -- DEBUGGING FUNCTION THAT DUMPS THE APU REGISTERS

(02031) * APDMP(V) -- DEBUGGING FUNCTION THAT DUMPS THE APU REGISTERS
(02032) * J. J. WOLF, 17 JULY 88
(02033) *
(02034) * BUFFER USAGE:
(02035) * V: COMPACT BUFFER, LENGTH 44 FULL WORDS, RECEIVES REGISTER CONTENTS
(02036) *
(02037) * APU PROGRAMS EMPTIES ALL REGISTERS INTO THE OQ
(02038) *
(02039) * EVEN
(02040) * DATA DUMPU$A
(02041) * DATA DUMPU$Z
(02042) *
(02043) * DUMP: BEGIN APU(DUMPU)
(02044) * DUMPU$A:
(02045) * ALL OUTPUTS ARE LEFT, THEN RIGHT
(02046) * OUTPUT R, A0-A7
(02047) * MOV(OQ),R(A0)
(02048) * MOV(OQ),R(A1)
(02049) * MOV(OQ),R(A2)
(02050) * MOV(OQ),R(A3)
(02051) * MOV(OQ),R(A4)
(02052) * MOV(OQ),R(A5)
(02053) * MOV(OQ),R(A6)
(02054) * MOV(OQ),R(A7)
(02055) * MOV(R,OQ)
(02056) * OUTPUT P, THEN 90 M0-M7, M1-M7, M2-M7, M3-M7 SO WE CAN (TRY TO)
(02057) * RECOVER THE CONTENTS OF M7, WHICH WE NEED TO CLOBBER
(02058) * MOV(OQ),MUL(M0,M7)
(02059) * MOV(OQ),MUL(M1,M7)
(02060) * MOV(OQ),MUL(M2,M7)
(02061) * MOV(OQ),MUL(M3,M7)
(02062) * MOV(P,OQ)
(02063) * NOW PUT 1.0 INTO M7 AND MULTIPLY TO OUTPUT M0-M3
(02064) * K(+1)
(02065) * MOV(R,M7)
(02066) * MUL(M0,M7)
(02067) * MOV(OQ),MUL(M1,M7)
(02068) * MOV(OQ),MUL(M2,M7)
(02069) * MOV(OQ),MUL(M3,M7)
(02070) * PUT 1.0 INTO M0 AND MULTIPLY TO OUTPUT M4-M6
(02071) * MOV(R,M0)
(02072) * MUL(M0,M4)
(02073) * MOV(OQ),MUL(M0,M5)
(02074) * MOV(OQ),MUL(M0,M6)
(02075) * MOV(P,OQ)
(02076) * FINALLY, OUTPUT EXI
(02077) * MOV(EXT,OQ)
(02078) * CLEAR(RA)
(02079) * JUMP(0)
(02080) * DUMPU$Z=RA-DUMPU$A
(02081) *
(02082) * END
(02083) * APS MODULE FOR APDMP

          * THAT'S ALL THERE IS TO SEE!

```

PAGE 661 (00ND) (00AL16) 00N16H.MSD.2, 29-DEC-80 14130:40, E0: WOLF
 APUMP(V) -- DEBUGGING FUNCTION THAT DUMPS THE APU REGISTERS

```

(02084) )
(02085) )NO INPUT STREAMI
(02086) ) NO: V(0),V(1),...V(43)
(02087) )
(02088) )
(02089) )PTR TO CONST. INSTR. BLOCK
(02090) )PTR TO SCALAR BLOCK
(02091) )NUMBER OF SCALARS
(02092) )MODULE SIZE IN HALFWORDS
(02093) )PTR TO CHAIN ANCHOR
(02094) )
(02095) )
(02096) )APS INPUT PROGRAM
(02097) )
(02098) )DUMPS: BEGIN APS(DUMPS)
(02099) )DUMPSI:
(02100) )JSH(P2,DUMPSO)
(02101) )SET(RO)
(02102) )CLEAR(R1)
(02103) )NOP(0)
(02104) )
(02105) )APS OUTPUT PROGRAM
(02106) )
(02107) )DUMPSO: SET(RA)
(02108) )LOAD(0W0,(0))
(02109) )LOAD(0W1,M$)
(02110) )SUB(0W0,M$)
(02111) )LOAD(0W1,45-1)
(02112) )ADD(0W0,(0),TF)
(02113) )SUBL(0W1,1),JUNPP(01)
(02114) )CLEAR(RO)
(02115) )NOP(0)
(02116) )END
(02117) )DUMPSA=PC
(02118) )DUMPSI: DATA 2F'0.0'
...
0000001E (02119) DUMPSZ=JL-DUMPS
  
```


PAGE 401 (00ND)(0CAL)00016N.050.2, 29-Dec-88 14:30:48, Fd1 WOLF
APU3 - WFC(V,U)

```

(02973) * COMPUTE E2 - MINIMUM MEAN-SQUARED PREDICTION ERROR
(02974) * STEP #1 - COMPUTE K1
(02975) * INPUT U(0) - STORE IN M6 AND A0 FOR SOLV
(02976) WFC$B1: MOV(IQ,M6)\NOP
(02977) * CALL SUBROUTINE TO GO SOLV OF U(0)
(02978) * 1/U(0) OUTPUT IN M6
(02979) * CALL(HPCUSDV)
(02980) * COMPUTE K1=U(1)/U(0)
(02981) * MOV(IQA,M5)\NOP
(02982) * MUL(M6,M5)\NOP
(02983) * STEP #2 - COMPUTE E1
(02984) * COMPUTE 1-K1**2
(02985) * K(1)\NOP
(02986) * MOV(R,A7)\NOP
(02987) * MOV(P,M7)\NOP
(02988) * MUL(P,M2)\NOP
(02989) * MUL(M2,M7)\NOP
(02990) * MOV(P,A3)\NOP
(02991) * SUB(A7,A3)\NOP
(02992) * MOV(R,M1)\NOP
(02993) * COMPUTE E1 = (1-K1**2)*U(0)
(02994) * MUL(M1,M6)\NOP
(02995) * STEP #3 - COMPUTE E2
(02996) * COMPUTE 1/E1 - 1/E1 IN M6
(02997) * MOV(P,A8)\NOP
(02998) * CALL(HPCUSDV)
(02999) * COMPUTE U(2)-K1*U(1)
(03000) * MUL(M2,M5)\NOP
(03001) * MOV(IQA,A2)\NOP
(03002) * MOV(P,A4)\NOP
(03003) * SUB(A2,A4)\NOP
(03004) * MOV(R,M7)\NOP
(03005) * COMPUTE K2=(U(2)-K1*U(1))/E1
(03006) * MUL(M6,M7)\NOP
(03007) * STEP #4 - COMPUTE E2
(03008) * COMPUTE K2**2
(03009) * MOV(P,M3)\NOP
(03010) * MOV(P,M5)\NOP
(03011) * MUL(M3,M5)\NOP
(03012) * MOV(P,A5)\NOP
(03013) * SUB(A7,A5)\NOP
(03014) * COMPUTE 1-K2**2
(03015) * SUB(A7,A5)\NOP
(03016) * MOV(R,M2)\NOP
(03017) * COMPUTE E2
(03018) * MUL(M2,M6)\NOP
(03019) * MOV(P,M3)\NOP
(03020) * END OF E2 COMPUTATION
(03021) *
(03022) * COMPUTE HIGH FREQUENCY CORRECTED
(03023) * AUTOCORRELATION COEFF'S AND OUTPUT
(03024) *
(03025) *

```


PAGE 71: [00ND1<0C116>00N16M.WS0.2, 29-Dec-88 14:38:48, Ed: WOLF
 APS3 - HFC(V,U,V)

A0C 06720 10AA002 (03124)	ADD(BR2,C103,TF)	JGEN V(1) ADDR
A0D 06722 1A0A002 (03125)	OUTPUT U(1),V(2) SEQUENCE	JGEN U(1) ADDR
A0E 06724 1C1A002 (03126)	ADD(BR0,C93,TF)	JGEN V(2) ADDR
A0F 06726 1E50003 (03127)	ADD(BR2,C103,TF)	JSET UP COUNTER
A10 06728 20A1004 (03128)	GEN REST OF U(1) ADDRS	JGEN U(1) ADDR
A11 0672A 22191001 (03129)	LOAD(BR1,3)	JDEC INDEX AND JUMP
A12 0672C 240A004 (03130)	WPCS\$3: ADD(BR0,C93,TF)	JGEN U(6) ADDR
	WPCS\$4: SUBL(BR1,1),JUMPP(HFCS\$3)	
	HPCS\$4: ADD(BR0,C93,TF)	
A13 0672E 2620031 (03133)	END OF APS INPUT PROGRAM	JHALT INPUT
A14 06730 2000020 (03134)	CLEAR(RI)	
	MOP(0)	
	(03137)	
	(03138)	
	(03139)	OUTPUT APS PROGRAM
	(03140)	REGISTER USAGE
	(03141)	BR0 Y ADDR
	(03142)	BR1 Y SIZE (UNUSED)
A15 06732 2A30032 (03143)	HPCS\$OP: SET(RA)	JTURN ON APU
	(03144)	
	(03145)	
A16 06734 2CC0000 (03146)	LOAD AND GEN Y ADDRS	JLOAD AND GEN Y(0)
A17 06736 2E50000 (03147)	HPCS\$5: LOAD(BR0,C03,TF)	JVS-1 (UNUSED)
A18 06738 300A000 (03148)	LOAD(BR1,M\$)	JGEN Y(1) ADDR
	ADD(BR0,M\$5,TF)	
A19 0673A 3250003 (03149)	GEN REST OF Y ADDRS	JSET UP COUNTER
A1A 0673C 340A000 (03150)	LOAD(BR1,3)	JGEN U(1) ADDR
A1B 0673E 36111A01 (03151)	HPCS\$6: ADD(BR0,C03,TF)	JDEC INDEX AND JUMP
A1C 06740 380A004 (03152)	SUBL(BR1,1),JUMPP(HFCS\$6)	JGEN Y(6) ADDR
	HPCS\$7: ADD(BR0,C03,TF)	
	(03155)	
A1D 06742 3A20030 (03156)	END OF APS OUTPUT PROGRAM	JHALT OUTPUT
A1E 06744 3C00020 (03157)	CLEAR(RO)	
	MOP(0)	
	(03159)	JCHAIN ANCHOR
	00006740 (03160)	HPCS\$A=BC
	(03161)	
06746	(03162)	END
	(03163)	
06746 0000000 (03164)	HPCS\$I DATA 14F'0.0'	
...		
0000005A (03165)	HPCS\$Z=BL-HFCS\$	JMODULE SIZE

PAGE 72: (RDMO) <DCA16> BBN16M.WSO.2, 29-Dec-88 14:38:48, Fdi: WOLF
APU3 - GAIN(Y,A,U,B,V,C)

```

(03166) * APU3 - GAIN(Y,A,U,B,V,C)
(03167) * WHR 9/88
(03168) *
(03169) * ROUTINE TO COMPUTE QUANTIZER SEGMENT
(03170) * SCALE FACTORS. DELTA GAINS AND OVERALL GAIN
(03171) * ARE ALSO COMPUTED AND QUANTIZED FOR YX.
(03172) *
(03173) * INPUTS:
(03174) * V(1),...,V(216) = SECOND RESIDUAL SAMPLES
(03175) * B = CONSTANT = 1/72
(03176) * C = CONSTANT = 1/3
(03177) *
(03178) * OUTPUTS:
(03179) * U(0),U(1),U(2) = CODED SEGMENT DELTA GAINS
(03180) * Y(0),Y(1),...,Y(5) = ALTERNATING SCALE FACTORS
(03181) * AND 1/SCALE FACTORS
(03182) * A = CODED OVERALL GAIN, G
(03183) *
PROCEDURE
  COMPUTE SUM(J), J=1,2,3
  SUM(J)=SUM(V(1+N*72)**2), I=1,...,72, N=0,1,2
  COMPUTE SG(J)=1/72*SUM(J), J=1,2,3
  COMPUTE G = OVERALL GAIN
  QUANTIZE G => GM - OUTPUT CODED G => SA
  COMPUTE CH*CH
  COMPUTE 1/(CH*CH)
  COMPUTE DG(1)=SG(1)/(CH*CH)
  QUANTIZE DG(1) => DGH(U) - OUTPUT CODED DG => U(0,1,2)
  COMPUTE SCALE FACTOR Y(1)=CH*DGH(1) => Y(0,2,4)
  COMPUTE 1/SCALE FACTOR => Y(1,3,5)

INPUT AND OUTPUT SEQUENCE
  IQ: SB,SC,V(1),...,V(72),CPW13,V(73),...,V(144),CPW13,
  V(145),...,V(216),CPW13,QUANT AND CODING TABLES
  FOR G, DG(1), DG(2), DG(3)
  OQ: SA,U(0),Y(0),Y(1),Y(2),Y(3),U(2),Y(4),Y(5)

APU REGISTER USAGE
  A0 SUM**2,HC**2,SDIV,Y(1)
  A1 SC(1)
  A2 SC(2)
  A3 SC(3)
  A4 SC(1)*SG(2),SDIV
  A5 QUANT
  A6 QUANT,SDIV
  A7 SUM**2,QUANT
  M0 SUM**2,SUM(SG(1)),CH,SDIV
  M1 SUM(1),SG(1),DGH(1)
  M2 SUM(2),SG(2),DGH(2)
  M3 SUM**2,SUM(3),SG(3),DGH(3)
  M4 SUM**2,SDIV
  M5 1/72,CH
  M6 SUM**2,CH**2,SDIV,Y(1)
  M7 1/3,1/(CH*CH)

GAIN - APU INITIALIZATION
  EVEN DATA GAIN$SA
  )START ON FULL WORD BOUNDARY
  )START ADDR AT SA

```

06762 0000

PAGE 73: (0000)<0CA16>00016M.WSD.2, 29-Dec-88 14:30:40, Ed: WMLF
APU3 - GAIN(V,A,M,R,N,C)

	06763 0004	(03219)	DATA	GAINSSZ	SIZE, COMPUTED AT END
	(03220)				
	(03221)		CAINS: BEGIN APO(CAIN)		
	(03222)		INPUT CONSTANTS 1/72 AND 1/3		
A00	06764 00E00000	(03223)	CAINS: MOV(IQA,M5) \ NOP		
A01	06766 00E00000	(03224)	MOV(IQA,M7) \ NOP		
	(03225)		COMPUTE SUM OF SQUARED SAMPLES		
	(03226)		SEGMENT #1		
A02	06768 30000061	(03227)	CALL(GAINSS)		
A03	0676A 20372037	(03228)	CLEAR(VI)		
A04	0676C 00090000	(03229)	MOV(R,M1) \ NOP		
	(03230)		SEGMENT #2		
A05	0676E 30000061	(03231)	CALL(GAINSS)		
A06	06770 20372037	(03232)	CLEAR(VI)		
A07	06772 000A0000	(03233)	MOV(R,M2) \ NOP		
	(03234)		SEGMENT #3		
A08	06774 30000061	(03235)	CALL(GAINSS)		
A09	06776 20372037	(03236)	CLEAR(VI)		
A0A	06778 00000000	(03237)	MOV(R,M3) \ NOP		
	(03238)		COMPUTE SC'S - SC(1)=1/72*SUM(1)		
	(03239)		SEGMENT #1		
A0B	0677A 04A00000	(03240)	MUL(M1,M5) \ NOP		
A0C	0677C 00010000	(03241)	MOV(P,A1) \ NOP		
A0D	0677E 00A90000	(03242)	MOV(P,M1) \ NOP		
	(03243)		SEGMENT #2		
A0E	06780 05200000	(03244)	MUL(M2,M5) \ NOP		
A0F	06782 00200000	(03245)	MOV(P,A2) \ NOP		
A10	06784 00AA0000	(03246)	MOV(P,M2) \ NOP		
	(03247)		SEGMENT #3		
A11	06786 05A00000	(03248)	MUL(M3,M5) \ NOP		
A12	06788 00030000	(03249)	MOV(P,A3) \ NOP		
A13	0678A 00AD0000	(03250)	MOV(P,M3) \ NOP		
	(03251)		COMPUTE OVERALL GAIN - G		
A14	0678C 42000000	(03252)	ADD(A1,A2) \ NOP		
A15	0678E 00940000	(03253)	MOV(R,A4) \ NOP		
A16	06790 44000000	(03254)	ADD(A3,A4) \ NOP		
A17	06792 00000000	(03255)	MOV(R,M0) \ NOP		
A18	06794 04000000	(03256)	MUL(M0,M7) \ NOP		
	(03257)		QUANTIZE OVERALL GAIN G = CH		
A19	06796 30000052	(03258)	CALL(GAINSSQ)		
A1A	06798 0000009C	(03259)	NOP \ MOV(R,00)		
	(03260)		COMPUTE GH**2		
A1B	0679A 00000000	(03261)	MOV(R,M0) \ NOP		
A1C	0679C 00000000	(03262)	MOV(R,M5) \ NOP		
A1D	0679E 04200000	(03263)	MUL(M0,M5) \ NOP		
	(03264)		COMPUTE 1/(GH*GH)		
A1E	067A0 00AF0000	(03265)	MOV(P,M6) \ NOP		
A1F	067A2 00000000	(03266)	MOV(P,M0) \ NOP		
A20	067A4 30000042	(03267)	CALL(GAINSDV)		
A21	067A6 00AF0000	(03268)	MOV(P,M7) \ NOP		
	(03269)		SEGMENT #1		
	(03270)		COMPUTE DC		
A22	067A8 04E00000	(03271)	MUL(M1,M7) \ NOP		

A23 067AA 30000052	(03272) * QUANTIZE DG	CALL(GAIN\$Q)	QUANTIZE DG
A24 067AC 0000009C	(03273) NOP \ MOV(R,0Q)	NOP \ MOV(R,0Q)	OUTPUT CODED DG
A25 067AE 00090000	(03274) MOV(R,M1) \ NOP	MOV(R,M1) \ NOP	M1=DCH(1)
A26 067B0 04A00000	(03275) * COMPUTE SCALE FACTOR V(1)	COMPUTE SCALE FACTOR V(1)	
A27 067B2 000C0000	(03276) MUL(M1,M5) \ NOP	MUL(M1,M5) \ NOP	P=DCH(1)*CH
A28 067B4 00AF0000	(03277) MOV(P,0Q) \ NOP	MOV(P,0Q) \ NOP	OUTPUT V(1)
A29 067B6 00000000	(03278) * COMPUTE 1/V(1)	COMPUTE 1/V(1)	
A2A 067B8 00AF0000	(03279) MOV(P,M6) \ NOP	MOV(P,M6) \ NOP	M6=V(1)
A2B 067BA 00000000	(03280) MOV(P,A0) \ NOP	MOV(P,A0) \ NOP	A0=V(1)
A2C 067BC 00000042	(03281) * CALL(GAIN\$DV)	CALL(GAIN\$DV)	
A2D 067BE 000C0000	(03282) MOV(P,0Q) \ NOP	MOV(P,0Q) \ NOP	SDIV - 1/V(1)
A2E 067BF 00000000	(03283) * SEGMENT #2		OUTPUT 1/V(1)
A2F 067C0 00000000	(03284) * COMPUTE DG	COMPUTE DG	
A30 067C2 000A0000	(03285) * QUANTIZE DG	QUANTIZE DG	
A31 067C4 05200000	(03286) MUL(M2,M7) \ NOP	MUL(M2,M7) \ NOP	P=SG(2)/CH**2
A32 067C6 00000000	(03287) * CALL(GAIN\$Q)	CALL(GAIN\$Q)	
A33 067C8 0000009C	(03288) NOP \ MOV(R,0Q)	NOP \ MOV(R,0Q)	QUANTIZE DG
A34 067CA 000A0000	(03289) MOV(R,M2) \ NOP	MOV(R,M2) \ NOP	OUTPUT CODED DG
A35 067CC 05200000	(03290) * COMPUTE SCALE FACTOR V(2)	COMPUTE SCALE FACTOR V(2)	
A36 067CE 000C0000	(03291) MUL(M2,M5) \ NOP	MUL(M2,M5) \ NOP	P=DCH(2)*CH
A37 067D0 00000000	(03292) MOV(P,0Q) \ NOP	MOV(P,0Q) \ NOP	OUTPUT V(2)
A38 067D2 00AF0000	(03293) * COMPUTE 1/V(2)	COMPUTE 1/V(2)	
A39 067D4 00000000	(03294) MOV(P,M6) \ NOP	MOV(P,M6) \ NOP	M6=V(2)
A3A 067D6 00000000	(03295) MOV(P,A0) \ NOP	MOV(P,A0) \ NOP	A0=V(2)
A3B 067D8 00000042	(03296) * CALL(GAIN\$DV)	CALL(GAIN\$DV)	
A3C 067DA 000C0000	(03297) MOV(P,0Q) \ NOP	MOV(P,0Q) \ NOP	SDIV - 1/V(2)
A3D 067DC 00000000	(03298) * SEGMENT #3		OUTPUT 1/V(2)
A3E 067DE 05E00000	(03299) * COMPUTE DG	COMPUTE DG	
A3F 067E0 00000052	(03300) * QUANTIZE DG	QUANTIZE DG	
A40 067E2 000A009C	(03301) CALL(GAIN\$Q)	CALL(GAIN\$Q)	
A41 067E4 00000000	(03302) NOP \ MOV(R,0Q)	NOP \ MOV(R,0Q)	QUANTIZE DG
A42 067E6 00000000	(03303) MOV(R,M3) \ NOP	MOV(R,M3) \ NOP	OUTPUT CODED DG
A43 067E8 05A00000	(03304) * COMPUTE SCALE FACTOR V(3)	COMPUTE SCALE FACTOR V(3)	
A44 067EA 000C0000	(03305) MUL(M3,M5) \ NOP	MUL(M3,M5) \ NOP	P=DCH(3)*CH
A45 067EC 00AF0000	(03306) MOV(P,0Q) \ NOP	MOV(P,0Q) \ NOP	OUTPUT V(3)
A46 067EE 00000000	(03307) * COMPUTE 1/V(3)	COMPUTE 1/V(3)	
A47 067F0 00000042	(03308) MOV(P,M6) \ NOP	MOV(P,M6) \ NOP	M6=V(3)
A48 067F2 000C0000	(03309) MOV(P,A0) \ NOP	MOV(P,A0) \ NOP	A0=V(3)
A49 067F4 20322032	(03310) * CALL(GAIN\$DV)	CALL(GAIN\$DV)	
A4A 067F6 00000000	(03311) MOV(P,0Q) \ NOP	MOV(P,0Q) \ NOP	SDIV - 1/V(3)
A4B 067F8 00000000	(03312) * CLEAR(RA)	CLEAR(RA)	OUTPUT 1/V(3)
A4C 067FA 00000000	(03313) * JUMP(0)	JUMP(0)	
A4D 067FC 00000000	(03314) * EJECT	EJECT	
A4E 067FE 00000000	(03315) * HALT APU	HALT APU	

```

PAGE 75: (00001)DC116>000100.2, 29-00c-00 14:30:40, Ed: WOLY
APU3 - GAIN(V,A,B,B,V,C)

(03323) * * * GAINSDV - DIVIDE SUBROUTINE - COMPUTES 1/C
(03324) * C IS ASSUMED TO BE IN M6 AND A0
(03325) *
A42 06700 16A00000 GAINSDV: R(2)\NOP
A43 0670A 00960000 MOV(R,A6)\NOP
A44 0670C 2E000000 RCP(A6)\NOP
A45 0670E 00000000 MOV(R,M6)\NOP
A46 06710 04400000 MUL(M6,M6)\NOP
A47 06712 00040000 MOV(P,A4)\NOP
A48 06714 4CC00000 SUB(A6,A4)\NOP
A49 06716 000C0000 MOV(R,M4)\NOP
A4A 06718 04000000 MUL(M6,M4)\NOP
A4B 0671A 00A00000 MOV(P,M6)\NOP
A4C 0671C 04400000 MUL(M6,M6)\NOP
A4D 0671E 00040000 MOV(P,A4)\NOP
A4E 06720 4CC00000 SUB(A6,A4)\NOP
A4F 06802 000C0000 MOV(R,M4)\NOP
A50 06804 04000000 MUL(M6,M4)\NOP
A51 06806 1000A000 RESULTS =1/C IN P1
      RETURN
      EJECT
(03342) *
(03343) *
(03344) *

```

```

R0=2.0
R16=2.0
R=1/C*0.001(-P0)
P=P0-C
R4=P0-C
R=2-P0-C
R4=R
P=P0*R(-P1)
R0=P1
P=P1-C
R4=P
R=2-P1-C
R4=R
P=P1*(2-P1-C)(-P2)

```

PAGE 76: (00ND1)DC116>00N16M.N50.2, 29-04c-00 14130:40, 2d1 VOL7
AP03 - GAIN(V,A,W,B,V,C)

```

(03345) * * * GAIN50 - QUANTIZATION SUBROUTINE
(03346) *
A52 06000 00000076 (03347) GAIN50: NOP \ MOV(IQA,A6)          JIMPUT 2*-13
A53 06004 00050015 (03348) * POT DATA IN A5 AND INIT CODE
A54 0600C 000002A0 (03349) * MOV(P,A5)\MOV(ZERO,A5)      JIMPUT DATA \ IMIT CODE=0
(03350) * MOV(A5,A5)
A55 0600E 00F60000 (03351) * INNER LOOP - FIND QUANT VALUE AND CODE
A56 06010 4F104085 (03352) GAIN50: MOV(IQA,A6)\NOP      JIMPUT BOUNDARY - TM(N)\NOP
(03353) * SUR(A5,A6)\MOV(A5),ADD(A5,A6)
A57 06012 00F70000 (03354) * MOV(IQA,A7)\NOP
A58 06014 00000000 (03355) * MOV(R,NULL)\NOP
A59 06016 0116005F (03356) * JUMPS(GAIN503,TM)
A5A 06018 001E0055 (03357) * JUMPC(GAIN501,T1)
(03358) * FUND0 THE RIGHT QUANTIZATION
A5B 0601A 00E000E0 (03359) * GAIN50: MOV(IQA,NULL)
A5C 0601C 00000000 (03360) * NOP
A5D 0601E 00160050 (03361) * JUMPC(GAIN502,TM1)
(03362) * TABLE FINISHED - OUTPUT DATA
A5E 06020 20372037 (03363) * GAIN503: CLEAR(01)
A5F 06022 02E03A00 (03364) * R(17)\ALIGN(A5)
(03365) * QUANTIZED VALUE IN R1 - CODE IN R2
A50 06024 10001000 (03366) * RETURN
(03367) *
(03368) *
(03369) *
EJECT

```


0606C (03423) • (03424) • END


```

(03563) * SPECIAL SUPPORT MODULES FOR FEOP & FEOT
(03564) * FORWARD & INVERSE EVEN-ODD SEPARATE MODULES.
(03565) * ALSO CALLED BY MOST FOR PPTMR AND PPTMR.
(03566) * (ALSO CALLED BY MOST FOR PPTLR AND PPTLR.)
(03567) * 15 MAY 79
(03568) * J F BURNS
(03569) *
(03570) * ENTER HERE FOR FEOT
(03571) *
(03572) *
(03573) * FEOPSSM MOVIR R6, PLSSET + PLSGO SET-GO LITERAL
(03574) * NOP FEOPSI & SAVE IT
(03575) * EVEN
(03576) *
(03577) * ENTER HERE FOR FEOP
(03578) *
(03579) * FEOPSSM MOVIR R6, PLSCLR + PLSGO CLR-GO LITERAL
(03580) * & SAVE IT
(03581) *
(03582) * COMMON PATH FOR FEOS SET-UP
(03583) *
(03584) * FEOPSI MOVIR R4, 4 * H$ (R1) GET W-ID
(03585) * ANDIR R4, MSKSLAY CHVT TO BCT INDEX
(03586) * LRS R4, 7
(03587) * EVEN
(03588) * MOVIR R5, DCT$AT + H$ (R4) GET PVR-OF-2 BS
(03589) * INCR R5, 1 CHVT TO REAL BS
(03590) * LLS R5, 2 CHVT TO COS TOL INDEX
(03591) * MOVIR R4, COS$ ADDR OF APS CONST
(03592) * PUSHMIL R4, SINCONST (R5) LOAD .SCOS(PI/N)
(03593) * PUSHMIL R4, SINCONST + W$ (R5) LOAD .SSIN(PI/N)
(03594) *
(03595) * CALL R1, AP$BNDR NORMAL BIND, BUT GO = 0
(03596) *
(03597) * MOVIR R6, AP$GO (R3) SET OR CLEAR GO CTRL
(03598) *
(03599) * RETURN
(03600) *
(03601) * EVEN

```

222

PAGE 04: (RUND)DCAL16>BWL16H.M50.2, 29-Dec-88 14:38:48, Ed: WOLF
APU3 - FEOSL FOURIER EVEN-ODD SEPARATE LONG (MAP-388)

```

A1A 06954 00F00F0 (03655)      MOV(TQA,A0)      A0=R(K)
A1B 06956 00F100F1 (03656)      MOV(TQA,A1)      A1=R(N-K)
A1C 06958 41004000 (03657)      ADD(A0,A1) \ SUB(A0,A1)  R=2UR\2VI
A1D 0695A 00F700F2 (03658)      MOV(TQA,A2)      A2=IK
A1E 0695C 00F300F3 (03659)      MOV(TQA,A3)      A3=I(N-K)
A1F 0695E 0510535 (03660)      MOV(A5),NUL(M2,M4) \ MOV(A5),NUL(M2,M5)
A20 06960 00940009 (03661) \ A5=S*SIN(K-1)\S*COS(K-1)
A21 06962 43504050 (03662)      MOV(R,A4) \ MOV(R,M1)  A4=2UR\2VI
A22 06964 00490054 (03663)      MOV(EXO),ADD(A2,A3) \ MOV(EXO),SUB(A2,A3)
A23 06966 00800097 (03664) \ EXO=2UR\2VI
A24 06968 00860086 (03665)      MOV(EXT,M1) \ MOV(EXT,A4)  M1=2VI\A4=2UR
A25 0696A 400R4500 (03666) \ MOV(R,M0) \ MOV(R,A7)  M0=2VR\A7=2VI
A26 0696C 00570040 (03667)      MOV(P,A6)      A6=C*COS(K-1)\C*SIN(K-1)
A27 0696E 008C0080 (03668)      MOV(EXO),SUB(A6,A5) \ MOV(EXO),ADD(A6,A5)
A28 06970 04000400 (03669) \ EXO=2VR\2VI
A29 06972 00980098 (03670) \ MOV(EXT,A7) \ MOV(EXT,M0)  A7=2VI\A0=2VR
A2A 06974 004D004C (03671) \ MOV(R,M4) \ MOV(R,M5)  M4=COS\A5=SIN
A2B 06976 00800080 (03672) \ NUL(M0,M4) \ NUL(M1,M5)  P=2VR*COS\2VI*SIN
A2C 06978 04300491 (03673) \ MOV(R,EXO) \ MOV(EXT,M4)  EXO=COS\A5=SIN
A2D 0697A 00510050 (03674) \ MOV(P,EXO)      M5=SIN\A4=COS
A2E 0697C 49004900 (03675) \ MOV(M0,M5) \ MOV(A1) \ MOV(M1,M4)
A2F 0697E 42924192 (03676) \ MOV(M0,A1) \ MOV(EXT,A0)  EXO=2VR*COS\2VI*SIN
A30 06980 00800081 (03677) \ SUB(A0,A1)      A0=2VR*COS\A1=2VI*SIN
A31 06982 00800080 (03678) \ MOV(A2),ADD(A4,A2) \ MOV(A2),SUB(A4,A2)  R=2VR*COS-2VI*SIN
A32 06984 00510050 (03679) \ MOV(P,A0) \ MOV(P,A1)  A2=2VR*COS-2VISIN
A33 06986 411C411C (03680) \ MOV(P,EXO)      A0=2VR*SIN\A1=2VI*COS
A34 06988 41P242P2 (03681) \ MOV(EXT,A1) \ MOV(EXT,A0)  EXO=2VR*SIN\2VICOS
A35 0698A 02520A52 (03682) \ MOV(OQ),ADD(A0,A1) \ MOV(OQ),ADD(A0,A1)  A1=2VICOS\A0=2VR*SIN
A36 0698C 009C009C (03683) \ MOV(A2),SUB(A7,A2) \ MOV(A2),ADD(A7,A2)  OUT=2VR(K)\2VI(K)
A37 0698E 901C0019 (03684) \ MOV(A2),R(A2) \ MOV(A2),NEG(A2)  A2=2VR*SIN*2VICOS
A38 06990 00000000 (03685) \ MOV(R,OQ) \ MOV(R,OQ)  A2=2VI(K)\-2VI(N-K)
A39 06992 10000059 (03686) \ JUMPC(FEOSL$2,EO)  OUT=2VI(K)\2VI(N-K)
A40 06994 05A00500 (03687) \ NOP
A41 06996 00F000F0 (03688) \ JUMPC(FEOSL$4)
A42 06998 00F100F1 (03689) \ MAIN LOOP FOR INVERSE (K=1,2,...,N/2)
A43 0699A 41004000 (03690) \ FEOSL$3
A44 0699C 00F200F2 (03691) \ MUL(M3,M5) \ MUL(M3,M4)  S*SIN(K-1)\S*COS(K-1)
A45 0699E 00F100F1 (03692) \ MOV(TQA,A0)      A0=R(K)
A46 0699F 41004000 (03693) \ MOV(TQA,A1) \ SUB(A0,A1)  A1=R(N-K)
A47 069A0 00F200F2 (03694) \ MOV(TQA,A2) \ MOV(TQA,A3)  R=2UR\2VI
A48 069A2 00F100F1 (03695) \ MOV(TQA,A2) \ MOV(TQA,A3)  A2=IK
A49 069A4 00F100F1 (03696) \ MOV(TQA,A2) \ MOV(TQA,A3)  A3=I(N-K)

```


PAGE 07: C00WD1C0CALC000W1C0.2, 29-Dec-80 14:30:40, E4: WOLF
APS3 - F11AL APS PROGRAM FOR FEUSL

```

A10 06A20 30320002 (03013) SUB(003,001) RESET TO REAL OF N-K
A10 06A22 32291302 (03014) SUBL(002,2),JUMPP(02) CONTINUE UNTIL LAST ELEMENT (N-2-2K-20)
A1A 06A24 34200031 (03016) CLEAR(01) HALT INPUT
A1B 06A26 36000020 (03018) NOP(0)
A1C 06A28 38300032 (03020) *OUTPUT PROGRAM
A1D 06A2A 3A403012 (03024) * F11AL33
A1E 06A2C 3C500000 (03025) SET(RA)
A1F 06A2E 3E600000 (03026) LOAD(000,C33)
A20 06A30 4010003A (03027) LOAD(001,M33)
A21 06A32 42310010 (03028) LOAD(002,M33)
A22 06A34 4431002A (03029) ADL(001,2)
A23 06A36 4631002A (03030) MOV(003,000)
A24 06A38 4831002A (03031) ADD(003,001)
A25 06A3A 4A31002A (03032) ADD(003,001)
A26 06A3C 4C0200A4 (03033) ADD(003,001)
A27 06A3E 4E0200A4 (03034) ADD(003,001)
A28 06A40 50000016 (03035) ADD(000,C113,TF)
A29 06A42 52000002 (03036) ADD(000,VR1,TF)
A2A 06A44 54020002 (03037) SUB(000,VR1)
A2B 06A46 5631003A (03038) SUBL(003,4)
A2C 06A48 58110032 (03039) SUBL(001,2)
A2D 06A4A 5A00000A (03040) *
A2E 06A4C 5C020002 (03041) *OUTPUT ADDRESS GENERATION LOOP
A2F 06A4E 5E0A0002 (03042) *
A30 06A50 60A00002 (03043) *
A31 06A52 62020002 (03044) *
A32 06A54 64370002 (03045) P4
A33 06A56 66120002 (03046) ADD(000,C113,TF)
A34 06A58 68200030 (03047) SUB(003,VR1,TF)
A35 06A5A 6A000020 (03048) ADD(003,VR1,TF)
A36 06A5C 6C000020 (03049) SUB(003,VR1)
A37 06A5E 6E000020 (03050) SUBL(001,2),JUMPP(P4)
A38 06A60 70000000 (03051) CLEAR(00)
A39 06A62 72000000 (03052) NOP(0)
A40 06A64 74000000 (03053) NOP(0)
A41 06A66 76000000 (03054) NOP(0)
A42 06A68 78000000 (03055) *
A43 06A6A 7A000000 (03056) *
A44 06A6C 7C000000 (03057) *
A45 06A6E 7E000000 (03058) F11AL3CA=JC ASSIGN A VALUE TO CHAIN ANCHOR
A46 06A70 80000000 (03059) *
A47 06A72 82000000 (03060) *
A48 06A74 84000000 (03061) *
A49 06A76 86000000 (03062) *
A50 06A78 88000000 (03063) *
A51 06A7A 8A000000 (03064) *
A52 06A7C 8C000000 (03065) *
A53 06A7E 8E000000 (03066) *
A54 06A80 90000000 (03067) *
A55 06A82 92000000 (03068) *
A56 06A84 94000000 (03069) *
A57 06A86 96000000 (03070) *
A58 06A88 98000000 (03071) *
A59 06A8A 9A000000 (03072) *
A60 06A8C 9C000000 (03073) *
A61 06A8E 9E000000 (03074) *
A62 06A90 A0000000 (03075) *
A63 06A92 A2000000 (03076) *
A64 06A94 A4000000 (03077) *
A65 06A96 A6000000 (03078) *
A66 06A98 A8000000 (03079) *
A67 06A9A AA000000 (03080) *
A68 06A9C AC000000 (03081) *
A69 06A9E AE000000 (03082) *
A70 06AA0 B0000000 (03083) *
A71 06AA2 B2000000 (03084) *
A72 06AA4 B4000000 (03085) *
A73 06AA6 B6000000 (03086) *
A74 06AA8 B8000000 (03087) *
A75 06AAA BA000000 (03088) *
A76 06AAC BC000000 (03089) *
A77 06AAE BE000000 (03090) *
A78 06AB0 C0000000 (03091) *
A79 06AB2 C2000000 (03092) *
A80 06AB4 C4000000 (03093) *
A81 06AB6 C6000000 (03094) *
A82 06AB8 C8000000 (03095) *
A83 06ABA CA000000 (03096) *
A84 06ABC CC000000 (03097) *
A85 06ABE CE000000 (03098) *
A86 06AB0 D0000000 (03099) *
A87 06AB2 D2000000 (03100) *
A88 06AB4 D4000000 (03101) *
A89 06AB6 D6000000 (03102) *
A90 06AB8 D8000000 (03103) *
A91 06ABA DA000000 (03104) *
A92 06ABC DC000000 (03105) *
A93 06ABE DE000000 (03106) *
A94 06AB0 E0000000 (03107) *
A95 06AB2 E2000000 (03108) *
A96 06AB4 E4000000 (03109) *
A97 06AB6 E6000000 (03110) *
A98 06AB8 E8000000 (03111) *
A99 06ABA EA000000 (03112) *
A100 06ABC EC000000 (03113) *
A101 06ABE EE000000 (03114) *
A102 06AB0 F0000000 (03115) *
A103 06AB2 F2000000 (03116) *
A104 06AB4 F4000000 (03117) *
A105 06AB6 F6000000 (03118) *
A106 06AB8 F8000000 (03119) *
A107 06ABA FA000000 (03120) *
A108 06ABC FC000000 (03121) *
A109 06ABE FE000000 (03122) *
A110 06AB0 00000000 (03123) *
A111 06AB2 02000000 (03124) *
A112 06AB4 04000000 (03125) *
A113 06AB6 06000000 (03126) *
A114 06AB8 08000000 (03127) *
A115 06ABA 0A000000 (03128) *
A116 06ABC 0C000000 (03129) *
A117 06ABE 0E000000 (03130) *
A118 06AB0 10000000 (03131) *
A119 06AB2 12000000 (03132) *
A120 06AB4 14000000 (03133) *
A121 06AB6 16000000 (03134) *
A122 06AB8 18000000 (03135) *
A123 06ABA 1A000000 (03136) *
A124 06ABC 1C000000 (03137) *
A125 06ABE 1E000000 (03138) *
A126 06AB0 20000000 (03139) *
A127 06AB2 22000000 (03140) *
A128 06AB4 24000000 (03141) *
A129 06AB6 26000000 (03142) *
A130 06AB8 28000000 (03143) *
A131 06ABA 2A000000 (03144) *
A132 06ABC 2C000000 (03145) *
A133 06ABE 2E000000 (03146) *
A134 06AB0 30000000 (03147) *
A135 06AB2 32000000 (03148) *
A136 06AB4 34000000 (03149) *
A137 06AB6 36000000 (03150) *
A138 06AB8 38000000 (03151) *
A139 06ABA 3A000000 (03152) *
A140 06ABC 3C000000 (03153) *
A141 06ABE 3E000000 (03154) *
A142 06AB0 40000000 (03155) *
A143 06AB2 42000000 (03156) *
A144 06AB4 44000000 (03157) *
A145 06AB6 46000000 (03158) *
A146 06AB8 48000000 (03159) *
A147 06ABA 4A000000 (03160) *
A148 06ABC 4C000000 (03161) *
A149 06ABE 4E000000 (03162) *
A150 06AB0 50000000 (03163) *
A151 06AB2 52000000 (03164) *
A152 06AB4 54000000 (03165) *
A153 06AB6 56000000 (03166) *
A154 06AB8 58000000 (03167) *
A155 06ABA 5A000000 (03168) *
A156 06ABC 5C000000 (03169) *
A157 06ABE 5E000000 (03170) *
A158 06AB0 60000000 (03171) *
A159 06AB2 62000000 (03172) *
A160 06AB4 64000000 (03173) *
A161 06AB6 66000000 (03174) *
A162 06AB8 68000000 (03175) *
A163 06ABA 6A000000 (03176) *
A164 06ABC 6C000000 (03177) *
A165 06ABE 6E000000 (03178) *
A166 06AB0 70000000 (03179) *
A167 06AB2 72000000 (03180) *
A168 06AB4 74000000 (03181) *
A169 06AB6 76000000 (03182) *
A170 06AB8 78000000 (03183) *
A171 06ABA 7A000000 (03184) *
A172 06ABC 7C000000 (03185) *
A173 06ABE 7E000000 (03186) *
A174 06AB0 80000000 (03187) *
A175 06AB2 82000000 (03188) *
A176 06AB4 84000000 (03189) *
A177 06AB6 86000000 (03190) *
A178 06AB8 88000000 (03191) *
A179 06ABA 8A000000 (03192) *
A180 06ABC 8C000000 (03193) *
A181 06ABE 8E000000 (03194) *
A182 06AB0 90000000 (03195) *
A183 06AB2 92000000 (03196) *
A184 06AB4 94000000 (03197) *
A185 06AB6 96000000 (03198) *
A186 06AB8 98000000 (03199) *
A187 06ABA 9A000000 (03200) *
A188 06ABC 9C000000 (03201) *
A189 06ABE 9E000000 (03202) *
A190 06AB0 A0000000 (03203) *
A191 06AB2 A2000000 (03204) *
A192 06AB4 A4000000 (03205) *
A193 06AB6 A6000000 (03206) *
A194 06AB8 A8000000 (03207) *
A195 06ABA AA000000 (03208) *
A196 06ABC AC000000 (03209) *
A197 06ABE AE000000 (03210) *
A198 06AB0 B0000000 (03211) *
A199 06AB2 B2000000 (03212) *
A200 06AB4 B4000000 (03213) *
A201 06AB6 B6000000 (03214) *
A202 06AB8 B8000000 (03215) *
A203 06ABA BA000000 (03216) *
A204 06ABC BC000000 (03217) *
A205 06ABE BE000000 (03218) *
A206 06AB0 C0000000 (03219) *
A207 06AB2 C2000000 (03220) *
A208 06AB4 C4000000 (03221) *
A209 06AB6 C6000000 (03222) *
A210 06AB8 C8000000 (03223) *
A211 06ABA CA000000 (03224) *
A212 06ABC CC000000 (03225) *
A213 06ABE CE000000 (03226) *
A214 06AB0 D0000000 (03227) *
A215 06AB2 D2000000 (03228) *
A216 06AB4 D4000000 (03229) *
A217 06AB6 D6000000 (03230) *
A218 06AB8 D8000000 (03231) *
A219 06ABA DA000000 (03232) *
A220 06ABC DC000000 (03233) *
A221 06ABE DE000000 (03234) *
A222 06AB0 E0000000 (03235) *
A223 06AB2 E2000000 (03236) *
A224 06AB4 E4000000 (03237) *
A225 06AB6 E6000000 (03238) *
A226 06AB8 E8000000 (03239) *
A227 06ABA EA000000 (03240) *
A228 06ABC EC000000 (03241) *
A229 06ABE EE000000 (03242) *
A230 06AB0 F0000000 (03243) *
A231 06AB2 F2000000 (03244) *
A232 06AB4 F4000000 (03245) *
A233 06AB6 F6000000 (03246) *
A234 06AB8 F8000000 (03247) *
A235 06ABA FA000000 (03248) *
A236 06ABC FC000000 (03249) *
A237 06ABE FE000000 (03250) *
A238 06AB0 00000000 (03251) *
A239 06AB2 02000000 (03252) *
A240 06AB4 04000000 (03253) *
A241 06AB6 06000000 (03254) *
A242 06AB8 08000000 (03255) *
A243 06ABA 0A000000 (03256) *
A244 06ABC 0C000000 (03257) *
A245 06ABE 0E000000 (03258) *
A246 06AB0 10000000 (03259) *
A247 06AB2 12000000 (03260) *
A248 06AB4 14000000 (03261) *
A249 06AB6 16000000 (03262) *
A250 06AB8 18000000 (03263) *
A251 06ABA 1A000000 (03264) *
A252 06ABC 1C000000 (03265) *
A253 06ABE 1E000000 (03266) *
A254 06AB0 20000000 (03267) *
A255 06AB2 22000000 (03268) *
A256 06AB4 24000000 (03269) *
A257 06AB6 26000000 (03270) *
A258 06AB8 28000000 (03271) *
A259 06ABA 2A000000 (03272) *
A260 06ABC 2C000000 (03273) *
A261 06ABE 2E000000 (03274) *
A262 06AB0 30000000 (03275) *
A263 06AB2 32000000 (03276) *
A264 06AB4 34000000 (03277) *
A265 06AB6 36000000 (03278) *
A266 06AB8 38000000 (03279) *
A267 06ABA 3A000000 (03280) *
A268 06ABC 3C000000 (03281) *
A269 06ABE 3E000000 (03282) *
A270 06AB0 40000000 (03283) *
A271 06AB2 42000000 (03284) *
A272 06AB4 44000000 (03285) *
A273 06AB6 46000000 (03286) *
A274 06AB8 48000000 (03287) *
A275 06ABA 4A000000 (03288) *
A276 06ABC 4C000000 (03289) *
A277 06ABE 4E000000 (03290) *
A278 06AB0 50000000 (03291) *
A279 06AB2 52000000 (03292) *
A280 06AB4 54000000 (03293) *
A281 06AB6 56000000 (03294) *
A282 06AB8 58000000 (03295) *
A283 06ABA 5A000000 (03296) *
A284 06ABC 5C000000 (03297) *
A285 06ABE 5E000000 (03298) *
A286 06AB0 60000000 (03299) *
A287 06AB2 62000000 (03300) *
A288 06AB4 64000000 (03301) *
A289 06AB6 66000000 (03302) *
A290 06AB8 68000000 (03303) *
A291 06ABA 6A000000 (03304) *
A292 06ABC 6C000000 (03305) *
A293 06ABE 6E000000 (03306) *
A294 06AB0 70000000 (03307) *
A295 06AB2 72000000 (03308) *
A296 06AB4 74000000 (03309) *
A297 06AB6 76000000 (03310) *
A298 06AB8 78000000 (03311) *
A299 06ABA 7A000000 (03312) *
A300 06ABC 7C000000 (03313) *
A301 06ABE 7E000000 (03314) *
A302 06AB0 80000000 (03315) *
A303 06AB2 82000000 (03316) *
A304 06AB4 84000000 (03317) *
A305 06AB6 86000000 (03318) *
A306 06AB8 88000000 (03319) *
A307 06ABA 8A000000 (03320) *
A308 06ABC 8C000000 (03321) *
A309 06ABE 8E000000 (03322) *
A310 06AB0 90000000 (03323) *
A311 06AB2 92000000 (03324) *
A312 06AB4 94000000 (03325) *
A313 06AB6 96000000 (03326) *
A314 06AB8 98000000 (03327) *
A315 06ABA 9A000000 (03328) *
A316 06ABC 9C000000 (03329) *
A317 06ABE 9E000000 (03330) *
A318 06AB0 A0000000 (03331) *
A319 06AB2 A2000000 (03332) *
A320 06AB4 A4000000 (03333) *
A321 06AB6 A6000000 (03334) *
A322 06AB8 A8000000 (03335) *
A323 06ABA AA000000 (03336) *
A324 06ABC AC000000 (03337) *
A325 06ABE AE000000 (03338) *
A326 06AB0 B0000000 (03339) *
A327 06AB2 B2000000 (03340) *
A328 06AB4 B4000000 (03341) *
A329 06AB6 B6000000 (03342) *
A330 06AB8 B8000000 (03343) *
A331 06ABA BA000000 (03344) *
A332 06ABC BC000000 (03345) *
A333 06ABE BE000000 (03346) *
A334 06AB0 C0000000 (03347) *
A335 06AB2 C2000000 (03348) *
A336 06AB4 C4000000 (03349) *
A337 06AB6 C6000000 (03350) *
A338 06AB8 C8000000 (03351) *
A339 06ABA CA000000 (03352) *
A340 06ABC CC000000 (03353) *
A341 06ABE CE000000 (03354) *
A342 06AB0 D0000000 (03355) *
A343 06AB2 D2000000 (03356) *
A344 06AB4 D4000000 (03357) *
A345 06AB6 D6000000 (03358) *
A346 06AB8 D8000000 (03359) *
A347 06ABA DA000000 (03360) *
A348 06ABC DC000000 (03361) *
A349 06ABE DE000000 (03362) *
A350 06AB0 E0000000 (03363) *
A351 06AB2 E2000000 (03364) *
A352 06AB4 E4000000 (03365) *
A353 06AB6 E6000000 (03366) *
A354 06AB8 E8000000 (03367) *
A355 06ABA EA000000 (03368) *
A356 06ABC EC000000 (03369) *
A357 06ABE EE000000 (03370) *
A358 06AB0 F0000000 (03371) *
A359 06AB2 F2000000 (03372) *
A360 06AB4 F4000000 (03373) *
A361 06AB6 F6000000 (03374) *
A362 06AB8 F8000000 (03375) *
A363 06ABA FA000000 (03376) *
A364 06ABC FC000000 (03377) *
A365 06ABE FE000000 (03378) *
A366 06AB0 00000000 (03379) *
A367 06AB2 02000000 (03380) *
A368 06AB4 04000000 (03381) *
A369 06AB6 06000000 (03382) *
A370 06AB8 08000000 (03383) *
A371 06ABA 0A000000 (03384) *
A372 06ABC 0C000000 (03385) *
A373 06ABE 0E000000 (03386) *
A374 06AB0 10000000 (03387) *
A375 06AB2 12000000 (03388) *
A376 06AB4 14000000 (03389) *
A377 06AB6 16000000 (03390) *
A378 06AB8 18000000 (03391) *
A379 06ABA 1A000000 (03392) *
A380 06ABC 1C000000 (03393) *
A381 06ABE 1E000000 (03394) *
A382 06AB0 20000000 (03395) *
A383 06AB2 22000000 (03396) *
A384 06AB4 24000000 (03397) *
A385 06AB6 26000000 (03398) *
A386 06AB8 28000000 (03399) *
A387 06ABA 2A000000 (03400) *
A388 06ABC 2C000000 (03401) *
A389 06ABE 2E000000 (03402) *
A390 06AB0 30000000 (03403) *
A391 06AB2 32000000 (03404) *
A392 06AB4 34000000 (03405) *
A393 06AB6 36000000 (03406) *
A394 06AB8 38000000 (03407) *
A395 06ABA 3A000000 (03408) *
A396 06ABC 3C000000 (03409) *
A397 06ABE 3E000000 (03410) *
A398 06AB0 40000000 (03411) *
A399 06AB2 42000000 (03412) *
A400 06AB4 44000000 (03413) *
A401 06AB6 46000000 (03414) *
A402 06AB8 48000000 (03415) *
A403 06ABA 4A000000 (03416) *
A404 06ABC 4C000000 (03417) *
A405 06ABE 4E000000 (03418) *
A406 06AB0 50000000 (03419) *
A407 06AB2 52000000 (03420) *
A408 06AB4 54000000 (03421) *
A409 06AB6 56000000 (03422) *
A410 06AB8 58000000 (03423) *
A411 06ABA 5A000000 (03424) *
A412 06ABC 5C000000 (03425) *
A413 06ABE 5E000000 (03426) *
A414 06AB0 60000000 (03427) *
A415 06AB2 62000000 (03428) *
A416 06AB4 64000000 (03429) *
A417 06AB6 66000000 (03430) *
A418 06AB8 68000000 (03431) *
A419 06ABA 6A000000 (03432) *
A420 06ABC 6C000000 (03433) *
A421 06ABE 6E000000 (03434) *
A422 06AB0 70000000 (03435) *
A423 06AB2 72000000 (03436) *
A424 06AB4 74000000 (03437) *
A425 06AB6 76000000 (03438) *
A426 06AB8 78000000 (03439) *
A427 06ABA 7A000000 (03440) *
A428 06ABC 7C000000 (03441) *
A429 06ABE 7E000000 (03442) *
A430 06AB0 80000000 (03443) *
A431 06AB2 82000000 (03444) *
A432 06AB4 84000000 (03445) *
A433 06AB6 86000000 (03446) *
A434 06AB8 88000000 (03447) *
A435 06ABA 8A000000 (03448) *
A436 06ABC 8C000000 (03449) *
A437 06ABE 8E000000 (03450) *
A438 06AB0 90000000 (03451) *
A439 06AB2 92000000 (03452) *
A440 06AB4 94000000 (03453) *
A441 06AB6 96000000 (03454) *
A442 06AB8 98000000 (03455) *
A443 06ABA 9A000000 (03456) *
A444 06ABC 9C000000 (03457) *
A445 06ABE 9E000000 (03458) *
A446 06AB0 A0000000 (03459) *
A447 06AB2 A2000000 (03460) *
A448 06AB4 A4000000 (03461) *
A449 06AB6 A6000000 (03462) *
A450 06AB8 A8000000 (03463) *
A451 06ABA AA000000 (03464) *
A452 06ABC AC000000 (03465) *
A453 06ABE AE000000 (03466) *
A454 06AB0 B0000000 (03467) *
A455 06AB2 B2000000 (03468) *
A456 06AB4 B4000000 (03469) *
A457 06AB6 B6000000 (03470) *
A458 06AB8 B8000000 (03471) *
A459 06ABA BA000000 (03472) *
A460 06ABC BC000000 (03473) *
A461 06ABE BE000000 (03474) *
A462 06AB0 C00000
```

PAGE 88: (CMDJ)<DC116>BNH16M-MS0.2, 29-Dec-88 14130140, Edt WOLF
APS3 - FILAL APS PROGRAM FOR PROSL

06A60 0000000 (03066) FILALSCI DATA 0F'0.0"

...

(03067) *
(03068) *
00000000 (03069) FILALSSZ= 0L-FILALS
(03070) *
(03071) *
(03072) *
(03073) *

COMPUTE MODULE SIZE

EVEN

```

(03074) - APU3 - APC(V,A,U,V,M,R,S,T) APC (WITH NOISE SHAPING)
(03075) - FIELD 0/00
(03076) -
(03077) - RUNS WITH APU3-APC (APC55)
(03078) -
(03079) - FUNCTION:
(03080) - COMPUTE CODED APC RESIDUAL FROM PREEMPHASIZED
(03081) - INPUT SPEECH. (SEE DETAILED DOCUMENTATION OF
(03082) - APC LOOP IN ALGORITHM SPECIFICATION.)
(03083) -
(03084) - PARAMETERS:
(03085) - IN: V(FIXED); CODED RESIDUAL SAMPLES
(03086) - IN: A(REAL); PITCH LAG (SA)
(03087) - IN: U(REAL); PREEMPHASIZED SPEECH
(03088) - IN: V(REAL); FILTER COEFFS: 6 SPECTRAL, 3 PITCH, 6 M.S.
(03089) - IN: W(REAL); 3 QUANTIZER SEGMENT SCALE FACTORS,
(03090) - ALTERNATING WITH THEIR RECIPROCAL,
(03091) - IN/OUT: R(REAL); PITCH FILTER HISTORY: LAST, CURRENT FRAMES
(03092) - IN/OUT: S(REAL); SPECTRAL FILTER HISTORY
(03093) - IN/OUT: T(REAL); NOISE SHAPING FILTER HISTORY
(03094) -
(03095) - SYMBOLS USED IN COMMENTS IN THIS MODULE:
(03096) - I: INDEX OF CURRENT ITERATION OF APC LOOP (01005-1)
(03097) - M: PITCH LAG (IN SAMPLES) (SA)
(03098) - ANS(1:16) NOISE SHAPING FILTER COEFFS (ZEROS) (V(9:14))
(03099) - AN(1:16) SPECTRAL FILTER (ALSO M.S. FILTER POLES) COEFFS (V(0:15))
(03100) - C1H PITCH FILTER COEFF #1 (V(6))
(03101) - C2H PITCH FILTER COEFF #2 (V(7))
(03102) - C3H PITCH FILTER COEFF #3 (V(8))
(03103) - Q1(1-6:I-1) NOISE SHAPING FILTER INPUT (T(0:15))
(03104) - V(1-6:I-1) SPECTRAL FILTER INPUT (S(0:15))
(03105) - RH(1-M+1) PITCH FILTER INPUT #1 (MOST RECENT) (RH(0)=R(FRMSZ)...
(03106) - RH(1-M) ...WHERE FRMSZ FIXED AT 216)
(03107) - RH(1-M-1) PITCH FILTER INPUT #2
(03108) - QP(1) M.S. FILTER (ZEROS) OUTPUT: SUMCANS(J)*Q1(1-J), J=1:6
(03109) - QP(1) M.S. FILTER (POLES) OUTPUT: SUMC AN(J)*Q1(1-J), J=1:6
(03110) - VHP(1) SPECTRAL FILTER OUTPUT: SUMC AN(J)*VH(1-J), J=1:6
(03111) - RHP(1) PITCH FILTER OUTPUT: C1H*RH(1-M+1) +
(03112) - C2H*RH(1-M) +
(03113) - C3H*RH(1-M-1)
(03114) - SP(1) SPEECH INPUT SAMPLE FOR CURRENT APC LOOP ITERATION (U(1))
(03115) - WO(1) UNQUANTIZED APC RESIDUAL: SP(1)*QP(1)-QP(1)*VHP(1)+RHP(1)
(03116) - GFAC(1:13) QUANTIZER SCALE FACTORS FOR FRAME SEGMENTS 1:3 (V(0,2,4))
(03117) - QD(1) NORMALIZED APC RESIDUAL: WO/GFAC
(03118) - LU(1) CODED APC RESIDUAL SAMPLE (V(1))
(03119) - WH(1) QUANTIZED APC RESIDUAL SAMPLE
(03120) - WH(1) SCALED QUANTIZED APC RESIDUAL SAMPLE: WH*GFAC
(03121) - Q(1) CURRENT SAMPLE QUANTIZATION NOISE: WH(1)-WO(1)
(03122) -
(03123) - THIS APU PROGRAM (IN CONJUNCTION WITH ITS APS PROGRAM)
(03124) - MODIFIES ONE INSTRUCTION IN ITS APS PROGRAM BY:
(03125) - 0. READING PITCH FROM SCALAR A, AND FIXING IT
(03126) - 1. WRITING IT INTO THE RH UP THE BUS-1 COPY OF THE INSTRUCTION

```

```

(03927) ) 2. WAITING ON OQE TO BE SURE IT'S THERE (PLOS 1 NOP THAT STORER
(03928) ) 3AYS IS NECESSARY)
(03929) ) 3. READING THE MODIFIED APS INSTRUCTION (FULLWORD) FROM BUS 1
(03930) ) 4. WRITING IT TO APSMEM IN PSEUDOMEMORY
(03931) ) 5. WAITING FOR OQE AGAIN BEFORE LETTING THE APS INPUT PROGRAM
(03932) ) PROCEED.
(03933) )
(03934) )
(03935) )
(03936) )
(03937) )
(03938) )
(03939) )
(03940) )
(03941) )
(03942) )
(03943) )
(03944) )
(03945) )
(03946) )
(03947) )
(03948) )
(03949) )
(03950) )
(03951) )
(03952) )
(03953) )
(03954) )
(03955) )
(03956) )
(03957) )
(03958) )
(03959) )
(03960) )
(03961) )
(03962) )
(03963) )
(03964) )
(03965) )
(03966) )
(03967) )
(03968) )
(03969) )
(03970) )
(03971) )
(03972) )
(03973) )
(03974) )
(03975) )
(03976) )
(03977) )
(03978) )
(03979) )

```

GENERAL SCHEME:

- 1) FIX PITCH, SEND TO APS PROGRAM
- 2) INITIALIZE REGISTERS FOR LOOP: GET VN'S, RN'S
- 3) APC LOOP:

COMPUTE FILTER OUTPUTS
 COMPUTE RESIDUAL: $WO = SP(I) + QP(I) - QP(I) + RHP(I) + VHP(I)$
 NORMALIZE RESIDUAL: $VO = WO/CFAC$
 CODE/QUANTIZE NORMALIZED RESIDUAL
 SCALE QUANTIZED RESIDUAL: $WN = WN + GPAC$
 UPDATE FILTER INPUTS: $Q(I) = WN(I) - WO(I)$
 $Q1(I) = Q(I) - QP(I)$
 $VN(I) = VN(I) - VHP(I)$
 $RN(I) = RN(I) - RHP(I)$
 DO NEXT APC LOOP ITERATION: $I = I + 1$

NOTE: SPECTRAL FILTER INPUTS AND PITCH FILTER INPUTS (VN'S & RN'S) ARE RETAINED FROM ITERATION TO ITERATION IN APS REGISTERS. W.S. FILTER INPUTS, AND ALL FILTER COEFFS, ARE RECALC FOR EACH ITERATION OF THE APC LOOP. W.S. FILTER INPUTS ARE ALSO UPDATED AND WRITTEN OUT AFTER EACH ITERATION'S USE.

REGISTER USAGE (FOR MAIN APC LOOP):

ADM1	ADM2
----	----
A0: RHP	A0: WO SUB/WD/Q
A1: RN(I-1)	A1: QP1
A2: RN(I-M)	A2: VHP
A3: RN(I-M-1)	A3: SCRATCH
A4: VN(I-1)	M0: ANS(2)/
A5: VN(I-3)	AH(2)
A6: VN(I-5)	M1: ANS(4)/
A7: SCRATCH	AH(4)
M0: ANS(1)/	M2: ANS(6)/
AH(1)/	M3: AN(6)
RN(I-M-1)	M4: WO/
C3H	UH/
M1: ANS(3)/	RN(I-M+1)
AH(3)	
M2: ANS(5)/	
AH(5)	
M3: RN(I-M)	
M4: Q1(I-2)/	
VN(I-1)/	
C3H	
M5: Q1(I-4)/	
VN(I-4)	
M6: Q1(I-6)/	
VN(I-6)	
M7: (1/GPAC)/	
CFAC/	
C1H	

!!! ALL BUFFERS MUST BE RE-LNG,CONTIGUOUS,
 !!! EXCEPT BUFFER Y WHICH MUST BE PRD,LNG,CNTG.

PAGE 94: C0000D1C0A16>00W16M.N50.2, 29-Dec-88 14:30:46, Ed: WOLF
AP03 - APC(V,A,U,V,W,R,S,T) APC (WITH NOISE SHAPING)

```

(04139) ) GET NEXT 2 AN'S
(04140) )
(04141) )
(04142) )
(04143) )
(04144) )
(04145) )
(04146) )
(04147) )
(04148) )
(04149) )
(04150) )
(04151) )
(04152) )
(04153) )
(04154) )
(04155) )
(04156) )
(04157) )
(04158) )
(04159) )
(04160) )
(04161) )
(04162) )
(04163) )
(04164) )
(04165) )
(04166) )
(04167) )
(04168) )
(04169) )
(04170) )
(04171) )
(04172) )
(04173) )
(04174) )
(04175) )
(04176) )
(04177) )
(04178) )
(04179) )
(04180) )
(04181) )
(04182) )
(04183) )
(04184) )
(04185) )
(04186) )
(04187) )
(04188) )
(04189) )
(04190) )
(04191) )
A2A 06AC6 00000009
A2B 06AC8 00000000
A2C 06ACA 00000003
A2D 06ACC 00000007
A2E 06ACE 00000760
A2F 06AD0 00000000
A30 06AD2 00000000
A31 06AD4 04030403
A32 06AD6 00004717
A33 06AD8 00000000
A34 06ADA 00000000
A35 06ADC 04170417
A36 06ADE 47604770
A37 06AE0 00000000
A38 06AE2 00000000
A39 06AE4 00070007
A3A 06AE6 47734773
A3B 06AE8 00000003
A3C 06AEA 00000007
A3D 06AEC 00000760
A3E 06AEE 00004911
A3F 06AF0 02000200
A40 06AF2 00000000
--- \ M1<=M(4)
M1<=M(3) \ ---
GET 2 W.S. ZEROS PARTIAL ACCUMS (PARTIAL QP'S) INTO ADM2,
ADD THEN, ADD RESULT TO W0 ACCUM.
MOV(R,EXO) \ MOV(R,A3)
NOP \ MOV(EI,I7)
NOP \ ADD(A3,A7)
DOING WITH ZEROS (STILL HAVE TO ADD QP(I) TO A0 (W0 ACCUM))
CONTINUE WITH POLES...
MOV \ MOV(IQA,M0)
MOV(IQI,M0) \ NOP
MOV(A3),MUL(M1,M5)
NOP \ MOV(A7),ADD(A0,A7)
NOP
NOP
MOV(A7),MUL(M0,M4)
ADD(A3,A7) \ MOV(A0),ADD(A3,A7)
MEANWHILE, GET 1ST 2 PITCH FILTER COEFFS...
NOP \ MOV(IQA,M7)
MOV(IQI,M7) \ NOP
MOV(P,A7)
MOV(A3),ADD(A3,A7)
GET 2 W.S. POLES PARTIAL ACCUMS (PARTIAL QP1'S) INTO ADM2,
ADD THEN, SAVE IN A1, SUBTRACT RESULT FROM W0 ACCUM.
MOV(R,EXO) \ MOV(R,A3)
NOP \ MOV(EI,I7)
NOP \ ADD(A3,A7)
NOP \ MOV(A1),SUB(A0,A1)
EXOK=PARTIAL QP1 \ A3<=PARTIAL QP1
--- \ A7<=PARTIAL QP1
--- \ PC=QP1(I)
--- \ A1<=QP1(I) R<=SP(I)+QP(I)-QP1(I)
START MOVING VM'S (SPECTRAL FILTER INPUTS) FROM A4,A5,A6 TO M4,M5,M6
AT SAME TIME, UPDATE VM'S IN A4,A5,A6 FOR NEXT ITERATION:
CURRENT VM(I-M) => NEXT ITER'S VM(I-(M+1))
R(A4) \ MOV(A0),R(A4)
NOP
R<=VM(I-1) \ R<=VM(I-2) A0<=SP+OP-QP1

```



```

PAGE 90: (BND)DCALCDBN16M.MS0.2, 29-Dec-88 14:38:48, Ed: WOLF
APU3 - APC(T,A,U,V,M,R,S,T) APC (WITH NOISE SHAPING)

--- \ A7<=PARTIAL VMP
--- \ RC=VMP(I)
--- \ A3<=VMP(I))RC=SP+QP-QP1+VMP

A5C 06024 00000057 (04245)
A5D 0602C 00004760 (04246)
A5E 0602E 00004212 (04247)
      MOV \ MOV(EXT,A7)
      MOV \ ADD(A3,A7)
      MOV \ ADD(A2),ADD(A0,A2)
      CONTINUE WITH PITCH FILTER..

A5F 06030 04130000 (04251)
      MOV(A3),MUL(M0,M4) \ MOV(P,EIO)
      P<=RN(I-M-1)C3M \ ---
      A7<=RN(I-M-1)C3M \ ---
      RC=SUN(1ST 2 PITCH PRODS) \
      AB<=SP(I)+QP(I)-QP1(I)+VMP(I)

A60 06032 00570000 (04253)
A61 06034 47600000 (04254)
      MOV(EXT,A7) \ MOP
      ADD(A3,A7) \ MOV(R,A0)

A62 06036 00000000 (04256)
A63 06038 00000000 (04258)
A64 0603A 00000000 (04260)
      MOP \ MOV(IA,M7)
      MOP
      MOP

A65 0603C 00070000 (04262)
A66 0603E 47730000 (04264)
A67 06040 00000000 (04266)
A68 06042 00000000 (04268)
A69 06044 00000000 (04270)
A6A 06046 00000000 (04272)
A6B 06048 00004760 (04274)
A6C 0604A 00000000 (04276)
A6D 0604C 00000000 (04278)
A6E 0604E 00000000 (04280)
A6F 06050 00000000 (04282)
      MOV(P,A7) \ MOP
      MOV(A3),ADD(A3,A7) \ MOP
      MOP
      MOV(R,EIO) \ MOP
      MOV(R,A0) \ MOV(EXT,A7)
      MOP \ ADD(A0,A7)
      MOP
      MOP \ MOV(R,A0)
      MOP \ MOV(R,M3)

A70 06052 00000050 (04284)
      MOP \ MUL(M3,M7)

NORMALIZE APC RESIDUAL:
-----
      CALCULATE UD (NORMALIZED RESIDUAL)
      BY MULTIPLYING UD BY (1/GFAC)

      MOP \ MUL(M3,M7)

CODE-DECODE NORMALIZED RESIDUAL:
-----
CODE AND QUANTIZE UD, GENERATING IU (CODED) & UH (QUANTIZED)

CODING TABLE (FOLDED BINARY CODE): (DON'T WORRY ABOUT BOUNDARY CONDITIONS)
      IU RANGE
      1 UD < -THRESH 1 VAL(1)
      2 -THRESH < UD < 0 0 VAL(2)
      3 0 < UD < THRESH 2 VAL(3)

```

[illegible]

PAGE 98: CERN03C0CA16>R0W16M.M50.2, 29-Dec-88 14:38:48, Ed: WOLV
APU) - APC(Y,A,V,W,R,S,T) APC (WITH NOISE SHAPING)

```

(04351) )
(04352) ) HERE FOR BIT 2
(04353) )
(04354) ) APC$B2:
APC 0608A 001C0000 MOV(ZERO,0Q) \ MOV(IQA,NULL)
APC 0608C 00000000 NOP \ MOV(IQA,M3)
APC 0608E 00000000 NOP \ MOV(IQA,NULL)
APC 06090 00000000 NOP \ MOV(IQA,NULL)
APC 06092 10000005 JUMP(APC$COM)

(04360) )
(04361) ) HERE FOR BIT 1
(04362) )
(04363) ) APC$B1:
APC 06094 009C0000 MOV(R,0Q) \ MOV(IQA,M3)
APC 06096 00000000 NOP \ MOV(IQA,NULL)
APC 06098 00000000 NOP \ MOV(IQA,NULL)
APC 0609A 00000000 NOP \ MOV(IQA,NULL)

(04368) )
(04369) )
(04370) ) HERE WHEN DONE WITH CODING AND QUANTIZATION
(04371) ) OH IS IN M3.
(04372) )
(04373) ) APC$COM:
(04374) )
(04375) )
(04376) ) SCALE THE QUANTIZED RESIDUAL:
(04377) ) -----
(04378) )
(04379) ) NOP \ MUL(M3,M7)

(04380) )
(04381) )
(04382) ) UPDATE ARRAYS:
(04383) ) -----
(04384) )
APC 0609E 00000007 NOP \ MOV(P,A7)
APC 060A0 00004000 NOP \ SUB(A7,A0)
APC 060A2 00004910 NOP \ MOV(A0),SUB(A0,A1)
APC 060A4 00004AFC NOP \ MOV(0Q),SUM(A7,A2)

APC 060A6 00000000 NOP \ MOV(R,END)
APC 060A8 00540000 MOV(EXT,A4) \ NOP
APC 060AA 40000000 SUB(A4,A0) \ NOP
APC 060AC 009C0000 MOV(R,0Q) \ NOP

(04394) )
(04395) )
(04396) ) DONE WITH APC LOOP.
(04397) ) LOOP TIL (PI)
(04398) )
APC 060AE 90100015 JUMPC(APC$TOP,FI)
APC 060B0 00000000 NOP

(04401) )
(04402) )
(04403) ) WHEN DONE, WRITE OUT VH'S

```


(88MB3) < DCAL16 > B0W16M.WS0.2, 29-Dec-88 14:30:40, Ed: WOLF
 APS3 - APC(V,A,U,V,W,R,S,T) APC (WITH NOISE SHAPING)

```

(04421) * APS3 - APC(V,A,U,V,W,R,S,T) APC (WITH NOISE SHAPING)
(04422) * KFIELD 9/88
(04423) *
(04424) * BINDING DONE BY SBH$APC TO ACCOMMODATE R, S & T BUFFERS
(04425) *
(04426) * !!! ALL BUFFERS MUST BE BL,LNG,CONTIGUOUS,
(04427) * !!! EXCEPT BUFFER Y WHICH MUST BE FID,LNG,CNTG.
(04428) *
(04429) * THIS APS PROGRAM (IN CONJUNCTION WITH ITS APU PROGRAM)
(04430) * MODIFIES ONE INSTRUCTION IN ITSELF BY:
(04431) * 0. READING PITCH FROM SCALAR A, AND FIXING IT
(04432) * 1. WRITING IT INTO THE RH OF THE BUS-1 COPY OF THE INSTRUCTION
(04433) * 2. WAITING ON ONE TO BE SURE IT'S THERE (PLUS 1 NOP THAT STORER
(04434) * SAYS IS NECESSARY)
(04435) * 3. READING THE MODIFIED APS INSTRUCTION (FULLWORD) FROM BUS 1
(04436) * 4. WRITING IT TO APSNEM IN PSEUDOMEMORY
(04437) * 5. WAITING FOR ONE AGAIN BEFORE LETTING THE APS INPUT PROGRAM
(04438) * PROCEED.
(04439) *
(04440) * INPUT SEQUENCE:
(04441) * SYMBOLS
(04442) * -----
(04443) * (2**15), PITCH,
(04444) * (MOD APS INSTR),
(04445) * RH(-M), RH(-M-1),
(04446) * VH(-6),...,VH(-1),
(04447) * LOOP PRMSZ (-216) TIMES:
(04448) * 1
(04449) * 0
(04450) * 1
(04451) * 1
(04452) * 1
(04453) * 1
(04454) * 1
(04455) * 1
(04456) * 1
(04457) * 1
(04458) * 1
(04459) * 1
(04460) * 1
(04461) * 1
(04462) * 1
(04463) * 1
(04464) * 1
(04465) * 1
(04466) * 1
(04467) * END OF LOOP
(04468) *
(04469) *
(04470) *
(04471) * OUTPUT SEQUENCE:
(04472) * SYMBOLS
(04473) * -----

```

```

(2**15), 3A,
(MOD APS INSTR),
R(PRMSZ-M), R(PRMSZ-M-1),
S(0),...,S(5),
U(1),
T(0),V(14),
...
T(5),V(9),
V(5),...,V(0),
V(6),V(7),V(8),
R(PRMSZ-I-M+1),
W(1) FOR ITERS 0:71
W(3) FOR ITERS 72:143
W(5) FOR ITERS 144:215,
(CODING THRESH),
(2**15),
W(0) FOR ITERS 0:71
W(2) FOR ITERS 72:143
W(4) FOR ITERS 144:215,
(QUANT VAL(1)),
(QUANT VAL(2)),
(QUANT VAL(3)),
(QUANT VAL(4)),
(LFI)

```

```

(2**15),
(QUANT VAL(1)),
(QUANT VAL(2)),
(QUANT VAL(3)),
(QUANT VAL(4)),
(LFI)

```

```

PAGE 1011 100ND10C0C10>00010010,29-Dec-88 1410:40, Ed: VOLV
          APS3 - APC(V,A,W,V,W,S,T) APC (WITH NOISE SHAPING)

(04474) ) (FID PITCH), (MOD APS INSTR), (FID PITCH), (MOD APS INSTR),
(04475) ) FID FMSZ (-216) TIMES!
(04476) ) 1 Q1(I-6),...,Q1(I-2), T(0),...,T(4),
(04477) ) 1 T(1), V(1),
(04478) ) 1 Q1(I-1), T(5),
(04479) ) 1 RH(1), R(FMSZ+1),
(04480) ) FID OF LOOP
(04481) ) V(1-6),...,V(1-1), S(0),...,S(5),
(04482) ) 1 Z(0)
(04483) )
(04484) )
(04485) ) DEFINE SYMBOLIC FRAMESIZE
(04486) )
          FMSZ = 216
00000000 (04487) )
          ZVEN
          ADDR 0 )PTR TO COMS INS BLOCK)
          ADDR 0 )SCALAR ADDRESS)
          DATA 0 )NUMBER OF SCALARS)
          DATA APCSZ )MODULE SIZE
          ADDR APCSSC )PTR TO CHAIN ANCHOR
          APCSZ
(04496) ) APCSZ
(04497) ) BEGIN APS(APCS)
          JSM(APCS$0,P2) )SET UP AND START OUTPUT PROGRAM
          SET(RO)
(04499) )
          ) INPUT PROGRAM
          BR0: SCRATCH
          BR1: PITCH / R(FMSZ+1-M+1)
          BR2: W(1)
          BR3: INPUT LOOP ITERATION COUNTER ((FMSZ-1)/0)
          APCSS1:
(04506) ) APCSS1:
(04507) )
(04508) )
(04509) ) DO INSTRUCTION MODIFICATION STUFF:
(04510) )
          LOAD(BR0,SVTSUM(1),TF) GEN (2**-15) ADDR
          APCSSA:
          LOAD(BR0,M$(1),TF) GEN SA ADDR
          SET(W1) WAIT FOR APS TO SIGNAL
          NOP WRITE COMPLETE
          LOAD(BR0,APCS$MOD(1),TF) GEN MOD INSTR ADDR
          SET(W1) WAIT FOR WRITE COMPLETE
          NOP
          APCSS1:
(04512) )
          APCSSA:
(04513) )
          APCSSA:
(04514) )
          APCSSA:
(04515) )
          APCSSA:
(04516) )
          APCSSA:
(04517) )
          APCSSA:
(04518) )
          APCSSA:
(04519) )
          APCSSA:
(04520) )
          APCSSA:
(04521) )
          APCSSA:
(04522) )
          APCSSA:
(04523) ) GEN R(FMSZ-M), R(FMSZ-M-1)
          APCSSA:
(04524) ) ALSO GET INITIAL R(FMSZ+1-M+1) (MINUS SPACING)
          APCSSA:
(04525) )
          APCSSA:
(04526) ) APCSS1:

```


PAGE 103: [DWD]<DCA16>DWDH16.M50.2, 29-Dec-00 14:30:40, Ed: WOLF
 AP33 - APC(V,A,U,V,N,R,S,T) APC (WITH NOISE SHAPING)

```

(04500) APC$IT3:
A20 06C10 40C00000 LOAD(BR0,M$$,TF) GEN ADDR T(3)
(04501) APC$V11:
A21 06C12 42C00000 LOAD(BR0,M$$,TF) GEN ADDR V(11)
(04503) APC$IT4:
A22 06C14 44C00000 LOAD(BR0,M$$,TF) GEN ADDR T(4)
(04505) APC$V10:
A23 06C16 46C00000 LOAD(BR0,M$$,TF) GEN ADDR V(10)
(04507) APC$IT5:
A24 06C18 48C00000 LOAD(BR0,M$$,TF) GEN ADDR T(5)
(04509) APC$V9:
A25 06C1A 41C00000 LOAD(BR0,M$$,TF) GEN ADDR V(9)
(04593) ) GEN V(5),...,V(0)
(04594) )
(04595) APC$V5:
A26 06C1C 4CC00000 LOAD(BR0,M$$,TF) GEN ADDR V(5)
A27 06C1E 4E200002 SUB(BR0,V$,TF) GEN ADDR V(4)
A28 06C20 50200002 SUB(BR0,V$,TF) GEN ADDR V(3)
A29 06C22 52020002 SUB(BR0,V$,TF) GEN ADDR V(2)
A2A 06C24 54020002 SUB(BR0,V$,TF) GEN ADDR V(1)
A2B 06C26 56020002 SUB(BR0,V$,TF) GEN ADDR V(0)
(04603) ) GEN V(6),V(7),V(8)
(04604) )
(04605) APC$V6:
A2C 06C28 58C00000 LOAD(BR0,M$$,TF) GEN ADDR V(6)
A2D 06C2A 5A000002 ADD(BR0,V$,TF) GEN ADDR V(7)
A2E 06C2C 5C000002 ADD(BR0,V$,TF) GEN ADDR V(8)
(04609) ) GEN R(FRMSZ+I-N+1)
(04610) )
(04611) )
A2F 06C2E 5E900002 ADD(P01,V$,TF) GEN ADDR R(FRMSZ+I-N+1)
(04613) )
(04614) ) FOR I=0:71 GEN: W(1),CODING THRESH,(2*-15),W(0)
(04615) ) I=72:143 : W(3),CODING THRESH,(2*-15),W(2)
(04616) ) I=144:215 : W(5),CODING THRESH,(2*-15),W(4)
(04617) )
(04618) ) I=0:71 => ITERATION COUNTER = BR3 = (FRMSZ-1):(FRMSZ-72) = (215:144)
(04619) ) I=72:143 => ITERATION COUNTER = BR3 = (FRMSZ-73):(FRMSZ-144) = (143:72)
(04620) ) I=144:215 => ITERATION COUNTER = BR3 = (FRMSZ-145):(FRMSZ-216) = (71:0)
(04621) ) SEE IF BR3 IS IN (215:144) => W(1)/W(0)
(04622) )
(04623) )
A30 06C30 60400007 LOAD(BR0,143)
A31 06C32 62037A7 SUBD(BR0,BR3),JUMPP(APCS$2) BR0 <= 143 - (ITER CTR)
(04626) ) JUMP IF BR3 NOT IN (215:144)
(04627) APC$W1:
A32 06C34 64C00000 LOAD(BR0,M$$,TF) GEN W(1)
A33 06C36 66C399FC LOAD(BR0,CTN0+W$,V$(1),TF) GEN POSITIVE VALUED CODING THRESH
A34 06C38 68C703CE LOAD(BR0,SVT$W1(1),TF) GEN (2*-15)
(04631) APC$W0:
A35 06C3A 6AC00000 LOAD(BR0,M$$,TF) GEN V(0)
  
```


(04766) • SONAPC SPECIAL BINDING MODULE FOR APC

(04819)	EVEN	R4,BCT\$BA(R6)	JBIT LEN., 2BIT BUS#, 17BIT BA.
(04820)	MOVNRL	R7,APCS\$+W\$-APCS\$V	JSET LEFT WORD OF INSTR
(04821)	MOVNR	R4,R7,\$FFF0	J'OR' IN HIGH 4 BITS
(04822)	TOKR	R4,R7,\$FFF0	JASSUME FID-LMG1 CLR WL BIT
(04823)	CDB	BITS\$B.R4	JMOVE PHORD TO BUS1
(04824)	MOVNRL	R4,APCS\$+W\$-APCS\$V	
(04825)	J		
(04826)	J		
(04827)	J		
(04828)	MOVNR	R6,2*W\$(R1)	JGET BID
(04829)	MOVER	R6,R6,WK\$LEFT	JCONVERT TO OCT INDEX
(04830)	LNS	R6,7	
(04831)	EVEN		
(04832)	MOVNRL	R4,BCT\$BA(R6)	JBIT LEN., 2BIT BUS#, 17BIT BA.
(04833)	MOVNR	R7,APCS\$+W\$-APCS\$V	JSET LEFT WORD OF INSTR
(04834)	TOKR	R4,R7,\$FFF0	J'OR' IN HIGH 4 BITS
(04835)	MOVNRL	R4,APCS\$+W\$-APCS\$V	JMOVE PHORD TO BUS1
(04836)	J		
(04837)	J		
(04838)	J		
(04839)	MOVNR	R6,2*W\$(R1)	JGET BID
(04840)	MOVER	R6,R6,WK\$RAVT	JCONVERT TO OCT INDEX
(04841)	LLS	R6,1	
(04842)	EVEN		
(04843)	J	V(5)... (TEMPORARILY PUT V(6) ADDR WHERE V(5) WILL GO)	
(04844)	MOVNRL	R4,BCT\$BA(R6)	JBIT LEN., 2BIT BUS#, 17BIT BA.
(04845)	MOVNR	R7,APCS\$+W\$-APCS\$V	JSET LEFT WORD OF INSTR
(04846)	TOKR	R4,R7,\$FFF0	J'OR' IN HIGH 4 BITS
(04847)	MOVNRL	R4,APCS\$+W\$-APCS\$V	JMOVE PHORD TO BUS1
(04848)	LAF	R7,APCS\$+W\$-APCS\$V	JSET ADDR OF BUFFER START
(04849)	ADDIR	R7,5*W\$	JADD HW OFFSET TO DESIRED SAMPLE
(04850)	SIF	R7,APCS\$+W\$-APCS\$V	JSTORE ADDR OF DESIRED SAMPLE
(04851)	J		
(04852)	J		
(04853)	MOVNRL	R4,BCT\$BA(R6)	JBIT LEN., 2BIT BUS#, 17BIT BA.
(04854)	MOVNR	R7,APCS\$+W\$-APCS\$V	JSET LEFT WORD OF INSTR
(04855)	TOKR	R4,R7,\$FFF0	J'OR' IN HIGH 4 BITS
(04856)	MOVNRL	R4,APCS\$+W\$-APCS\$V	JMOVE PHORD TO BUS1
(04857)	LAF	R7,APCS\$+W\$-APCS\$V	JSET ADDR OF BUFFER START
(04858)	ADDIR	R7,6*W\$	JADD HW OFFSET TO DESIRED SAMPLE
(04859)	SIF	R7,APCS\$+W\$-APCS\$V	JSTORE ADDR OF DESIRED SAMPLE
(04860)	J		
(04861)	J		
(04862)	MOVNRL	R4,BCT\$BA(R6)	JBIT LEN., 2BIT BUS#, 17BIT BA.
(04863)	MOVNR	R7,APCS\$+W\$-APCS\$V	JSET LEFT WORD OF INSTR
(04864)	TOKR	R4,R7,\$FFF0	J'OR' IN HIGH 4 BITS
(04865)	MOVNRL	R4,APCS\$+W\$-APCS\$V	JMOVE PHORD TO BUS1
(04866)	LAF	R7,APCS\$+W\$-APCS\$V	JSET ADDR OF BUFFER START
(04867)	ADDIR	R7,9*W\$	JADD HW OFFSET TO DESIRED SAMPLE
(04868)	SIF	R7,APCS\$+W\$-APCS\$V	JSTORE ADDR OF DESIRED SAMPLE
(04869)	J		
(04870)	J		
(04871)	J		
(04872)	MOVNRL	R4,BCT\$BA(R6)	JBIT LEN., 2BIT BUS#, 17BIT BA.

0600E 0706C16 (04072)	MOVNR	R7,APC\$+W\$-APC\$V10	GET LEFT WORD OF INSTR
06010 564FF7F (04073)	IORER	R4,R7,\$FF7F	FOR IN HIGH 4 BITS
06012 84406C16 (04074)	MOVNML	R4,APC\$+W\$-APC\$V10	MOVE WORD TO BUS1
06014 87706C16 (04075)	LAF	R7,APC\$+W\$-APC\$V10	GET ADDR OF BUFFER START
06016 9C700014 (04076)	ADDR	R7,10-W\$	ADD HW OFFSET TO DESIRED SAMPLE
06018 87706C16 (04077)	SAP	R7,APC\$+W\$-APC\$V10	STORE ADDR OF DESIRED SAMPLE
06019 87706C16 (04078)			
0601A 87706C16 (04079)			
0601B 87706C16 (04080)			
0601C 87706C16 (04081)	MOVNML	R4,BCT\$BA(R6)	18BIT LEN., 28BIT BUS, 17BIT BA.
0601E 564FF7F (04082)	MOVNR	R7,APC\$+W\$-APC\$V11	GET LEFT WORD OF INSTR
06020 84406C12 (04083)	IORER	R4,R7,\$FF7F	FOR IN HIGH 4 BITS
06022 87706C12 (04084)	MOVNML	R4,APC\$+W\$-APC\$V11	MOVE WORD TO BUS1
06024 9C700016 (04085)	LAF	R7,APC\$+W\$-APC\$V11	GET ADDR OF BUFFER START
06026 87706C12 (04086)	ADDR	R7,11-W\$	ADD HW OFFSET TO DESIRED SAMPLE
06028 87706C12 (04087)	SAP	R7,APC\$+W\$-APC\$V11	STORE ADDR OF DESIRED SAMPLE
06029 87706C12 (04088)			
0602A 87706C12 (04089)			
0602C 564FF7F (04090)	MOVNML	R4,BCT\$BA(R6)	18BIT LEN., 28BIT BUS, 17BIT BA.
0602E 84406C0E (04091)	MOVNR	R7,APC\$+W\$-APC\$V12	GET LEFT WORD OF INSTR
06030 87706C0E (04092)	IORER	R4,R7,\$FF7F	FOR IN HIGH 4 BITS
06032 87706C0E (04093)	MOVNML	R4,APC\$+W\$-APC\$V12	MOVE WORD TO BUS1
06034 9C700018 (04094)	LAF	R7,APC\$+W\$-APC\$V12	GET ADDR OF BUFFER START
06036 87706C0E (04095)	ADDR	R7,12-W\$	ADD HW OFFSET TO DESIRED SAMPLE
06038 87706C0E (04096)	SAP	R7,APC\$+W\$-APC\$V12	STORE ADDR OF DESIRED SAMPLE
06039 87706C0E (04097)			
0603A 87706C0E (04098)			
0603C 84406C0A (04099)	MOVNML	R4,BCT\$BA(R6)	18BIT LEN., 28BIT BUS, 17BIT BA.
0603E 87706C0A (04100)	MOVNR	R7,APC\$+W\$-APC\$V13	GET LEFT WORD OF INSTR
06040 9C70001A (04101)	IORER	R4,R7,\$FF7F	FOR IN HIGH 4 BITS
06042 87706C0A (04102)	MOVNML	R4,APC\$+W\$-APC\$V13	MOVE WORD TO BUS1
06044 87706C0A (04103)	LAF	R7,APC\$+W\$-APC\$V13	GET ADDR OF BUFFER START
06046 87706C0A (04104)	ADDR	R7,13-W\$	ADD HW OFFSET TO DESIRED SAMPLE
06048 87706C0A (04105)	SAP	R7,APC\$+W\$-APC\$V13	STORE ADDR OF DESIRED SAMPLE
06049 87706C0A (04106)			
0604A 87706C0A (04107)			
0604C 84406C06 (04108)	MOVNML	R4,BCT\$BA(R6)	18BIT LEN., 28BIT BUS, 17BIT BA.
0604E 87706C06 (04109)	MOVNR	R7,APC\$+W\$-APC\$V14	GET LEFT WORD OF INSTR
06050 9C70001C (04110)	IORER	R4,R7,\$FF7F	FOR IN HIGH 4 BITS
06052 87706C06 (04111)	MOVNML	R4,APC\$+W\$-APC\$V14	MOVE WORD TO BUS1
06054 9C70001C (04112)	LAF	R7,APC\$+W\$-APC\$V14	GET ADDR OF BUFFER START
06056 87706C06 (04113)	ADDR	R7,14-W\$	ADD HW OFFSET TO DESIRED SAMPLE
06058 87706C06 (04114)	SAP	R7,APC\$+W\$-APC\$V14	STORE ADDR OF DESIRED SAMPLE
06059 87706C06 (04115)			
0605A 87706C06 (04116)			
0605B 87706C06 (04117)			
0605C 87706C06 (04118)			
0605D 87706C06 (04119)			
0605E 87706C06 (04120)			
0605F 87706C06 (04121)			
06060 87706C06 (04122)			
06061 87706C06 (04123)			
06062 87706C06 (04124)			
06063 87706C06 (04125)			
06064 87706C06 (04126)			
06065 87706C06 (04127)			
06066 87706C06 (04128)			
06067 87706C06 (04129)			
06068 87706C06 (04130)			
06069 87706C06 (04131)			
06070 87706C06 (04132)			
06071 87706C06 (04133)			
06072 87706C06 (04134)			
06073 87706C06 (04135)			
06074 87706C06 (04136)			
06075 87706C06 (04137)			
06076 87706C06 (04138)			
06077 87706C06 (04139)			
06078 87706C06 (04140)			
06079 87706C06 (04141)			
06080 87706C06 (04142)			
06081 87706C06 (04143)			
06082 87706C06 (04144)			
06083 87706C06 (04145)			
06084 87706C06 (04146)			
06085 87706C06 (04147)			
06086 87706C06 (04148)			
06087 87706C06 (04149)			
06088 87706C06 (04150)			
06089 87706C06 (04151)			
06090 87706C06 (04152)			
06091 87706C06 (04153)			
06092 87706C06 (04154)			
06093 87706C06 (04155)			
06094 87706C06 (04156)			
06095 87706C06 (04157)			
06096 87706C06 (04158)			
06097 87706C06 (04159)			
06098 87706C06 (04160)			
06099 87706C06 (04161)			
06100 87706C06 (04162)			
06101 87706C06 (04163)			
06102 87706C06 (04164)			
06103 87706C06 (04165)			
06104 87706C06 (04166)			
06105 87706C06 (04167)			
06106 87706C06 (04168)			
06107 87706C06 (04169)			
06108 87706C06 (04170)			
06109 87706C06 (04171)			
06110 87706C06 (04172)			
06111 87706C06 (04173)			
06112 87706C06 (04174)			
06113 87706C06 (04175)			
06114 87706C06 (04176)			
06115 87706C06 (04177)			
06116 87706C06 (04178)			
06117 87706C06 (04179)			
06118 87706C06 (04180)			
06119 87706C06 (04181)			
06120 87706C06 (04182)			
06121 87706C06 (04183)			
06122 87706C06 (04184)			
06123 87706C06 (04185)			
06124 87706C06 (04186)			
06125 87706C06 (04187)			
06126 87706C06 (04188)			
06127 87706C06 (04189)			
06128 87706C06 (04190)			
06129 87706C06 (04191)			
06130 87706C06 (04192)			
06131 87706C06 (04193)			
06132 87706C06 (04194)			
06133 87706C06 (04195)			
06134 87706C06 (04196)			
06135 87706C06 (04197)			
06136 87706C06 (04198)			
06137 87706C06 (04199)			
06138 87706C06 (04200)			
06139 87706C06 (04201)			
06140 87706C06 (04202)			
06141 87706C06 (04203)			
06142 87706C06 (04204)			
06143 87706C06 (04205)			
06144 87706C06 (04206)			
06145 87706C06 (04207)			
06146 87706C06 (04208)			
06147 87706C06 (04209)			
06148 87706C06 (04210)			
06149 87706C06 (04211)			
06150 87706C06 (04212)			
06151 87706C06 (04213)			
06152 87706C06 (04214)			
06153 87706C06 (04215)			
06154 87706C06 (04216)			
06155 87706C06 (04217)			
06156 87706C06 (04218)			
06157 87706C06 (04219)			
06158 87706C06 (04220)			
06159 87706C06 (04221)			
06160 87706C06 (04222)			
06161 87706C06 (04223)			
06162 87706C06 (04224)			
06163 87706C06 (04225)			
06164 87706C06 (04226)			
06165 87706C06 (04227)			
06166 87706C06 (04228)			
06167 87706C06 (04229)			
06168 87706C06 (04230)			
06169 87706C06 (04231)			
06170 87706C06 (04232)			
06171 87706C06 (04233)			
06172 87706C06 (04234)			
06173 87706C06 (04235)			
06174 87706C06 (04236)			
06175 87706C06 (04237)			
06176 87706C06 (04238)			
06177 87706C06 (04239)			
06178 87706C06 (04240)			
06179 87706C06 (04241)			
06180 87706C06 (04242)			
06181 87706C06 (04243)			
06182 87706C06 (04244)			
06183 87706C06 (04245)			
06184 87706C06 (04246)			
06185 87706C06 (04247)			
06186 87706C06 (04248)			
06187 87706C06 (04249)			
06188 87706C06 (04250)			
06189 87706C06 (04251)			
06190 87706C06 (04252)			
06191 87706C06 (04253)			
06192 87706C06 (04254)			
06193 87706C06 (04255)			
06194 87706C06 (04256)			
06195 87706C06 (04257)			
06196 87706C06 (04258)			
06197 87706C06 (04259)			
06198 87706C06 (04260)			
06199 87706C06 (04261)			
06200 87706C06 (04262)			
06201 87706C06 (04263)			
06202 87706C06 (04264)			
06203 87706C06 (04265)			
06204 87706C06 (04266)			
06205 87706C06 (04267)			
06206 87706C06 (04268)			
06207 87706C06 (04269)			
06208 87706C06 (04270)			
06209 87706C06 (04271)			
06210 87706C06 (04272)			
06211 87706C06 (04273)			
06212 87706C06 (04274)			
06213 87706C06 (04275)			
06214 87706C06 (04276)			
06215 87706C06 (04277)			
06216 87706C06 (04278)			
06217 87706C06 (04279)			
06218 87706C06 (04280)			
06219 87706C06 (04281)			
06220 87706C06 (04282)			
06221 87706C06 (04283)			
06222 87706C06 (04284)			
06223 87706C06 (04285)			
06224 87706C06 (04286)			
06225 87706C06 (04287)			
06226 87706C06 (04288)			
06227 87706C06 (04289)			
06228 87706C06 (04290)			
06229 87706C06 (04291)			
06230 87706C06 (04292)			
06231 87706C06 (04293)			
06232 87706C06 (04294)			
06233 87706C06 (04295)			
06234 87706C06 (04296)			
06235 87706C06 (04297)			
06236 87706C06 (04298)			
06237 87706C06 (04299)			
06238 87706C06 (04300)			
06239 87706C06 (04301)			
06240 87706C06 (04302)			
06241 87706C06 (04303)			
06242 87706C06 (04304)			
06243 87706C06 (04305)			
06244 87706C06 (04306)			
06245 87706C06 (04307)			
06246 87706C06 (04308)			
06247 87706C06 (04309)			
06248 87706C06 (04310)</			


```

0605C 564FF7F0 (04925) TORER R4,R7,5FFF0
0605E 84406C3A (04926) MOVNRL R4,APCS$+W$-APCS$W0
                                W(1)...
06060 C04C0502 (04928) MOVNRL R4,BCT$BA(R6)
06062 F0706C34 (04929) MOVNRL R7,APCS$+W$-APCS$W1
06064 564FF7F0 (04930) TORER R4,R7,5FFF0
06066 84406C34 (04931) MOVNRL R4,APCS$+W$-APCS$W1
06068 EF706C34 (04932) MOVNRL R4,APCS$+W$-APCS$W1
0606A 9C700002 (04933) LAF R7,1*W$
0606C EF706C34 (04934) ADDIR R7,APCS$+W$-APCS$W1
                                W(2)...
0606E C04C0502 (04936) MOVNRL R4,BCT$BA(R6)
06070 F0706C40 (04937) MOVNRL R7,APCS$+W$-APCS$W2
06072 564FF7F0 (04938) TORER R4,R7,5FFF0
06074 84406C40 (04939) MOVNRL R4,APCS$+W$-APCS$W2
06076 EF706C40 (04940) LAF R7,APCS$+W$-APCS$W2
06078 9C700004 (04941) ADDIR R7,2*W$
0607A EF706C40 (04942) SAF R7,APCS$+W$-APCS$W2
                                W(3)...
0607C C04C0502 (04944) MOVNRL R4,BCT$BA(R6)
0607E F0706C42 (04945) MOVNRL R7,APCS$+W$-APCS$W3
06080 564FF7F0 (04946) TORER R4,R7,5FFF0
06082 84406C42 (04947) MOVNRL R4,APCS$+W$-APCS$W3
06084 EF706C42 (04948) LAF R7,APCS$+W$-APCS$W3
06086 9C700006 (04949) ADDIR R7,3*W$
06088 EF706C42 (04950) SAF R7,APCS$+W$-APCS$W3
                                W(4)...
0608A C04C0502 (04952) MOVNRL R4,BCT$BA(R6)
0608C F0706C52 (04953) MOVNRL R7,APCS$+W$-APCS$W4
0608E 564FF7F0 (04954) TORER R4,R7,5FFF0
06090 84406C52 (04955) MOVNRL R4,APCS$+W$-APCS$W4
06092 EF706C52 (04956) LAF R7,APCS$+W$-APCS$W4
06094 9C700008 (04957) ADDIR R7,4*W$
06096 EF706C52 (04958) SAF R7,APCS$+W$-APCS$W4
                                W(5)...
06098 C04C0502 (04960) MOVNRL R4,BCT$BA(R6)
0609A F0706C4C (04961) MOVNRL R7,APCS$+W$-APCS$W5
0609C 564FF7F0 (04962) TORER R4,R7,5FFF0
0609E 84406C4C (04963) MOVNRL R4,APCS$+W$-APCS$W5
060A0 EF706C4C (04964) LAF R7,APCS$+W$-APCS$W5
060A2 9C70000A (04965) ADDIR R7,5*W$
060A4 EF706C4C (04966) SAF R7,APCS$+W$-APCS$W5
                                R BASE BINDING
060A6 F0620003 (04971) MOVNRL R6,3*HS(R1)
060A8 506C00FF (04972) MOVNRL R6,R6,MSK$RBYT
060AA 3A61 (04973) LLS R6,1
                                JGET BID
                                JCONVERT TU OCT INDEX

```

```

06DAB 0000 (04970) )
              (04979) )
              ) R(0) (INPUT)...
06DAC C04C0502 (04981) ) MOVNRL R4,BCT$BA(R6)
06DAE F0706022 (04982) ) MOVNR R7,APCS$+W$-APCS$18
06DAB 564E7F70 (04983) ) TORR R4,R7,$FF70
06D02 04406022 (04984) ) MOVNRL R4,APCS$+W$-APCS$18
              (04985) )
06D04 C04C0502 (04986) ) R(0) (OUTPUT)...
06D06 F0706C70 (04987) ) MOVNRL R4,BCT$BA(R6)
06D08 564E7F70 (04988) ) MOVNR R7,APCS$+W$-APCS$08
06D0A 04406C70 (04989) ) TORR R4,R7,$FF70
              (04990) ) MOVNRL R4,APCS$+W$-APCS$08
              (04991) )
              ) S BASE BINDING
              (04992) )
              (04993) )
06DBC F0620004 (04994) ) MOVNR R6,4*W$ (R1)
06D0E 506CFF70 (04995) ) MOVNR R6,R6,MSK$RBYT
06DC0 3C67 (04996) ) LRS R6,7
06DC1 0000 (04997) ) EVEN
              (04998) )
              ) S(0) (INPUT)...
06DC2 C04C0502 (05000) ) MOVNRL R4,BCT$BA(R6)
06DC4 F07060F0 (05001) ) MOVNR R7,APCS$+W$-APCS$18
06DC6 564E7F70 (05002) ) TORR R4,R7,$FF70
06DC8 044060F0 (05003) ) MOVNRL R4,APCS$+W$-APCS$18
              (05004) )
              ) S(0) (OUTPUT)...
06DCA C04C0502 (05005) ) MOVNRL R4,BCT$BA(R6)
06DCC F0706C08 (05006) ) MOVNR R7,APCS$+W$-APCS$08
06DCE 564E7F70 (05007) ) TORR R4,R7,$FF70
06DD0 04406C08 (05008) ) MOVNRL R4,APCS$+W$-APCS$08
              (05009) )
              ) T BASE BINDING
              (05010) )
              (05011) )
              (05012) )
06DD2 F0620004 (05013) ) MOVNR R6,4*W$ (R1)
06DD4 506C00F7 (05014) ) MOVNR R6,R6,MSK$RBYT
06DD6 3A61 (05015) ) LLS R6,1
06DD7 0000 (05016) ) EVEN
              (05017) )
              ) T(0) (INPUT)...
06DD8 C04C0502 (05018) ) MOVNRL R4,BCT$BA(R6)
06DDA F0706C04 (05019) ) MOVNR R7,APCS$+W$-APCS$18
06DDC 564E7F70 (05020) ) TORR R4,R7,$FF70
06DDE 04406C04 (05021) ) MOVNRL R4,APCS$+W$-APCS$18
              (05022) )
              ) T(1) (INPUT)...
              (05023) )
              (05024) )
06DF0 C04C0502 (05025) ) MOVNRL R4,BCT$BA(R6)
06DE2 F0706C08 (05026) ) MOVNR R7,APCS$+W$-APCS$18
06DE4 564E7F70 (05027) ) TORR R4,R7,$FF70
06DE6 04406C08 (05028) ) MOVNRL R4,APCS$+W$-APCS$18
06DE8 E7706C08 (05029) ) LAR R7,APCS$+W$-APCS$18
06DEA 9C700002 (05030) ) ADDR R7,1*W$

```

18BIT LEN., 28BIT BUS0, 170BIT BA.
GET LEFT WORD OF INSTR
'OR' IN HIGH 4 BITS
MOVE WORD TO BUS1

18BIT LEN., 28BIT BUS0, 170BIT BA.
GET LEFT WORD OF INSTR
'OR' IN HIGH 4 BITS
MOVE WORD TO BUS1

GET BID
CONVERT TO OCT INDEX

18BIT LEN., 28BIT BUS0, 170BIT BA.
GET LEFT WORD OF INSTR
'OR' IN HIGH 4 BITS
MOVE WORD TO BUS1

18BIT LEN., 28BIT BUS0, 170BIT BA.
GET LEFT WORD OF INSTR
'OR' IN HIGH 4 BITS
MOVE WORD TO BUS1

GET BID
CONVERT TO OCT INDEX

18BIT LEN., 28BIT BUS0, 170BIT BA.
GET LEFT WORD OF INSTR
'OR' IN HIGH 4 BITS
MOVE WORD TO BUS1

18BIT LEN., 28BIT BUS0, 170BIT BA.
GET LEFT WORD OF INSTR
'OR' IN HIGH 4 BITS
MOVE WORD TO BUS1
GET ADDR OF BUFFER START
ADD HV OFFSET TO DESIRED SAMPLE

06D0C	EF706C08	(05031)	SAP	R7,APCS\$W\$*APCS\$1T1	STORE ADDR OF DESIRED SAMPLE
		(05032)			
		(05033)	T(2)	(INPUT)...	
06D0E	C04C0502	(05034)	MOVRL	R4,ACT\$BA(R6)	18BIT LEN., 2BIT BUS#, 17BIT BA.
06D0F	F0706C0C	(05035)	MOVRL	R7,APCS\$W\$*APCS\$1T2	GET LEFT WORD OF INSTR
06D0G	564FF7F0	(05036)	IORR	R4,R7,\$FF70	OR- IN HIGH 4 BITS
06D0H	84406C0C	(05037)	MOVRL	R4,APCS\$W\$*APCS\$1T2	MOVE WORD TO BUS1
06D0I	8E706C0C	(05038)	LAF	R7,APCS\$W\$*APCS\$1T2	GET ADDR OF BUFFER START
06D0J	9C70000A	(05039)	ADDIR	R7,3-W\$	ADD HW OFFSET TO DESIRED SAMPLE
06D0K	EF706C0C	(05040)	SAP	R7,APCS\$W\$*APCS\$1T2	STORE ADDR OF DESIRED SAMPLE
		(05041)			
		(05042)	T(3)	(INPUT)...	
06D0C	C04C0502	(05043)	MOVRL	R4,ACT\$BA(R6)	18BIT LEN., 2BIT BUS#, 17BIT BA.
06D0E	F0706C10	(05044)	MOVRL	R7,APCS\$W\$*APCS\$1T3	GET LEFT WORD OF INSTR
06D0F	564FF7F0	(05045)	IORR	R4,R7,\$FF70	OR- IN HIGH 4 BITS
06D0G	84406C10	(05046)	MOVRL	R4,APCS\$W\$*APCS\$1T3	MOVE WORD TO BUS1
06D0H	8E706C10	(05047)	LAF	R7,APCS\$W\$*APCS\$1T3	GET ADDR OF BUFFER START
06D0I	9C70000A	(05048)	ADDIR	R7,3-W\$	ADD HW OFFSET TO DESIRED SAMPLE
06D0J	EF706C10	(05049)	SAP	R7,APCS\$W\$*APCS\$1T3	STORE ADDR OF DESIRED SAMPLE
		(05050)			
		(05051)	T(4)	(INPUT)...	
06E0A	C04C0502	(05052)	MOVRL	R4,ACT\$BA(R6)	18BIT LEN., 2BIT BUS#, 17BIT BA.
06E0C	F0706C14	(05053)	MOVRL	R7,APCS\$W\$*APCS\$1T4	GET LEFT WORD OF INSTR
06E0E	564FF7F0	(05054)	IORR	R4,R7,\$FF70	OR- IN HIGH 4 BITS
06E0F	84406C14	(05055)	MOVRL	R4,APCS\$W\$*APCS\$1T4	MOVE WORD TO BUS1
06E10	8E706C14	(05056)	LAF	R7,APCS\$W\$*APCS\$1T4	GET ADDR OF BUFFER START
06E12	9C70000A	(05057)	ADDIR	R7,4-W\$	ADD HW OFFSET TO DESIRED SAMPLE
06E14	EF706C14	(05058)	SAP	R7,APCS\$W\$*APCS\$1T4	STORE ADDR OF DESIRED SAMPLE
		(05059)			
		(05060)	T(5)	(INPUT)...	
06E18	C04C0502	(05061)	MOVRL	R4,ACT\$BA(R6)	18BIT LEN., 2BIT BUS#, 17BIT BA.
06E1A	F0706C18	(05062)	MOVRL	R7,APCS\$W\$*APCS\$1T5	GET LEFT WORD OF INSTR
06E1C	564FF7F0	(05063)	IORR	R4,R7,\$FF70	OR- IN HIGH 4 BITS
06E1E	84406C18	(05064)	MOVRL	R4,APCS\$W\$*APCS\$1T5	MOVE WORD TO BUS1
06E20	8E706C18	(05065)	LAF	R7,APCS\$W\$*APCS\$1T5	GET ADDR OF BUFFER START
06E22	9C70000A	(05066)	ADDIR	R7,5-W\$	ADD HW OFFSET TO DESIRED SAMPLE
06E24	EF706C18	(05067)	SAP	R7,APCS\$W\$*APCS\$1T5	STORE ADDR OF DESIRED SAMPLE
		(05068)			
		(05069)	T(6)	(OUTPUT)...	
06E26	C04C0502	(05070)	MOVRL	R4,ACT\$BA(R6)	18BIT LEN., 2BIT BUS#, 17BIT BA.
06E28	F0706C76	(05071)	MOVRL	R7,APCS\$W\$*APCS\$0T0	GET LEFT WORD OF INSTR
06E2A	564FF7F0	(05072)	IORR	R4,R7,\$FF70	OR- IN HIGH 4 BITS
06E2C	84406C76	(05073)	MOVRL	R4,APCS\$W\$*APCS\$0T0	MOVE WORD TO BUS1
		(05074)			
06E2E	0000FF63	(05075)	JMP	AP\$ENDRO	

PAGE 113: (00NDJ)C0CA16200N16M.MSD.2, 29-Dec-80 14:30:40, ED: VOLFF
APU3 - GTHR(V,A,B,V,W)

```

(05076) * APJ3 - GTHR(V,A,U,B,V,W)
(05077) * KFIELD 9/80
(05078) *
(05079) * RUNS WITH GTHR$S
(05080) *
(05081) * FUNCTION:
(05082) *   GATHER CODED TRANSMISSION DATA INTO OUTPUT BUFFER Y.
(05083) *   Y <=
(05084) *     ( 1) CODED APC RESIDUAL SAMPLES
(05085) *     ( 3) CODED ENERGY
(05086) *     ( 3) CODED DELTA GAINS
(05087) *     ( 1) CODED PITCH LAG
(05088) *     ( 3) CODED PITCH PREDICTOR COEFFS
(05089) *     ( 6) CODED SPECTRAL PREDICTOR COEFFS
(05090) * -----
(05091) * (230)

```

ON FUNCTION CALL, V(9-215) IS EXPECTED TO CONTAIN
216 CODED APC RESIDUAL SAMPLES.
ALL PARAMETERS (BUFFERS & SCALARS) ARE TREATED AS FWD, LNC
(FOR REAL SCALARS, THIS IMPLIES USING THE LEFT WORD OF
THE 32 BIT SCALAR.)

```

V(216)          <= SA
V(217-219)    <= U(4-2)
V(220)        <= SB
V(221-223)    <= V(4-2)
V(224-229)    <= W(4-5)

```

INPUT SEQUENCE:

$$u(n), \dots, u(5), [p1]$$

OUTPUT SEQUENCE:

 $y(216), \dots, y(229), [20]$

МЭК 3

DATA	GTHRU\$A
DATA	GTHRU\$Z

ВЕСИМ АРУ(СТНРУ)

!ASAMH GTHBUSA:

MOV(101,00) \ NOP
JUMPC(6THRUSSA,EN)

CLEAR(RA)
JUMP(0)

END
= 0A-GTHRUSSA

```

;COPY INPUT TO OUTPUT
;LOOP TIL DONE

```

ADDRESS	OPERATION	OPERANDS	REMARKS
05125	AP3 - CTRN(V,A,U,B,V,W)		
05126	KFIELD 9/80		
05127			
05128	INPUT SEQUENCE:		
05129	SA, U(0), U(1), U(2),		
05130	SB, V(0), V(1), V(3),		
05131	W(0), ..., W(5), [P1]		
05132			
05133	OUTPUT SEQUENCE:		
05134	V(216), ..., V(229), [EO]		
05135			
05136	ALL DATA ITEMS ARE FWD,LNG (TE FORMAT)		
05137			
05138	EVEN		
05139	ADDR	CTHRS\$1	PTR TO CONS INS BLOCK
05140	ADDR	CTHRS\$+2*CTHRS\$	SCALAR BLOCK
05141	DATA	2	# OF SCALARS
05142	DATA	CTHRS\$2	MODULE SIZE
05143	ADDR	CTHRS\$A	PTR TO CHAIN ANCHOR
05144			
05145	CTHRS\$: BEGIN APS(CTHRS)		
05146	JSM(CTHRS\$0,P2)		
05147	SET(RO)		
05148			
05149	INPUT PCM		
05150			
05151	CTHRS\$:		
05152	LOAD(BR0,M\$\$(1),TE)		
05153	LOAD(BR0,M\$\$(1))		
05154			
05155	GEN U(0-2)		
05156			
05157	LOAD(BR1,C13,TE)		
05158	LOAD(BR2,M\$)		
05159	LOAD(BR2,M\$)		
05160			
05161	ADD(BR1,C97,TE)		
05162	ADD(BR1,C97,TE)		
05163			
05164	GEN SB		
05165			
05166	MOV(BR0,BR0,TE)		
05167			
05168	GEN V(0-2)		
05169			
05170	LOAD(BR1,C23,TE)		
05171	LOAD(BR2,M\$)		
05172	LOAD(BR2,M\$)		
05173			
05174	ADD(BR1,C107,TE)		
05175	ADD(BR1,C107,TE)		
05176			
05177	GEN W(0-5)		

PAGE 115: (00001)MC16>00016M.N50.2, 29-Dec-88 14:38:48, Ed: MOLF
AP33 - CTRN(Y,A,W,B,V,W)

107 06260 18503002	(05170) ;		
110 04662 20600000	(05179)	LOAD(RR1,C3)	LOAD W(0) ADDR
111 04664 22120000	(05100)	LOAD(RR2,M5)	JUMPMV
	(05101)	SUB(RR1,M5)	LOAD W(-1) ADDR
112 04666 24600005	(05102) ;	LOAD(RR2,6-1)	JGEN 6 SAMPLES
113 04668 271A0000	(05103)	ADD(RR1,C113,TE)	JGEN W ELEMENTY ADDR
114 0466A 20291301	(05104) 811	SUBC(RR2,1),JUMPP(01)	
	(05105)		
	(05106) ;	CLEAR(R1)	DONE WITH INPUT
115 0466C 2A200031	(05107)	NOF(0)	
116 0466E 2C000020	(05108)	EJECT	
	(05109)		

PAGE 116: (0000) <DC116> 000106.M50.2, 79-Dec-80 14:30:40, Edt VOLP
 IPS3 - CTR(Y,A,W,B,V,W)

```

(05190) ;
(05191) ; OUTPUT PCM
(05192) ;
(05193) CTRSS01
A17 06E70 2E300032 (05194) SET(RA)
(05195) ;
(05196) ; GEN Y(216-229)
(05197) ;
A18 06E72 3050000A (05198) LOAD(0W1,C03)
A19 06E74 32600000 (05199) LOAD(0W2,M$)
A1A 06E76 34120000 (05200) SUB(0W1,M$)
(05201) ;
A1B 06E78 36110000 (05202) ADD(0W1,216)
A1C 06E7A 30600000 (05203) LOAD(0W2,19-1)
(05204) ;
A1D 06E7C 3A11000A (05205) #2: ADD(0W1,C03,TE)
A1E 06E7E 3C211001 (05206) SUBL(0W2,1),JUMPF(02)
(05207) ;
A1F 06E80 3E200030 (05208) CLEAR(RO)
A20 06E82 40000020 (05209) NOP(0)
(05210) ;
000067C (05211) CTRSSA-JC
(05212) ;
06E84 (05213) END
(05214) ;
06E84 00000000 (05215) CTRSS1 DATA 10P-0.0"
...
(05216) ;
0000056 (05217) CTRSS2=BL-CTRSS
MODULE SIZE

```

```

PAGE 117: (000010C0A16)00016M.N50.2; 29-Dec-88 14:38:48, Ed: VOLF
APU3 - THIST(V,U,V,W,R,S) SET UP INITTER FRAME HISTORIES
(05218) ' APU3 - THIST(V,U,V,W,R,S) SET UP INITTER FRAME HISTORIES
(05219) ' RTIELD 7/88
(05220) '
(05221) ' RUNS WITH APU3-THIST (THSS)
(05222) '
(05223) ' Y(0)-Y(WBS-1) <= U(0)-U(WBS-1)
(05224) ' V(0)-V(WBS-1) <= W(0)-W(WBS-1)
(05225) ' R(0)-R(WBS-1) <= S(0)-S(WBS-1)
(05226) '
(05227) ' (REQUIRES SPECIAL BINDING TO ACCOMMODATE R & S BUFFERS)
(05228) '
(05229) ' INPUT SEQUENCE:
(05230) ' U(0),...,U(WBS-1),
(05231) ' V(0),...,V(WBS-1),
(05232) ' S(0),...,S(WBS-1), EPI3
(05233) '
(05234) ' OUTPUT SEQUENCE:
(05235) ' Y(0),...,Y(WBS-1),
(05236) ' V(0),...,V(WBS-1),
(05237) ' R(0),...,R(WBS-1), (EO)
(05238) '
(05239) ' EVEN
(05240) ' DATA THUS$A
(05241) ' DATA THUS$Z
(05242) '
(05243) ' THUS$ BEGIN APU3(THISTW)
(05244) ' THUS$A: MOV(TQ1,OQ) \ NOP
(05245) ' JUMPC(THUS$A,EO)
(05246) '
(05247) ' CLEAR(MA)
(05248) ' JUMP(0)
(05249) ' THUS$Z = BA-THUS$A
(05250) ' END

```


PAGE 110: (BAND)DCAL16>BBI6M.MSO.2, 29-Dec-88 14:30:40, Ed: WOLF
APS3 - THIST(Y,V,W,R,S) APS PROGRAM FOR THIST

```

(05251) APS3 - THIST(Y,V,W,R,S) APS PROGRAM FOR THIST
(05252) / RFIELD 7/80
(05253) /
(05254) / BINDING DONE BY SEMSTM TO ACCOMMODATE R & S BUFFERS
(05255) /
(05256) / INPUT SEQUENCE:
(05257) / U(0),...,U(WBS-1),
(05258) / V(0),...,V(WBS-1),
(05259) / S(0),...,S(RBS-1), [PI]
(05260) /
(05261) / OUTPUT SEQUENCE:
(05262) / Y(0),...,Y(WBS-1),
(05263) / V(0),...,V(WBS-1),
(05264) / R(0),...,R(RBS-1), [EO]
(05265) /
(05266) /
(05267) / ADDR 0 / (PTR TO CONS INS BLOCK)
(05268) / ADDR 0 / (SCALAR ADDRESS)
(05269) / DATA 0 / (NUMBER OF SCALARS)
(05270) / DATA THSZ / (MODULE SIZE)
(05271) / ADDR THSZ / (PTR TO CHAIN ANCHOR)
(05272) /
(05273) THSZ BEGIN APS(THIST)
(05274) JSN(THSZ,0,72) / SET UP AND START OUTPUT PROGRAM
(05275) SET(RO)
(05276) /
(05277) / INPUT PROGRAM
(05278) / BR0: SAMPLE ADDRESSES
(05279) / BR1: BUFFER SIZES
(05280) /
(05281) THSZ:
(05282) / DO W BUFFER...
(05283) / THSZBR0:
(05284) / THSZBR1:
(05285) / THSZBR2:
(05286) / THSZBR3:
(05287) /
(05288) / THSZBR4:
(05289) /
(05290) /
(05291) / DO W BUFFER...
(05292) / THSZBR4:
(05293) / THSZBR5:
(05294) / THSZBR6:
(05295) / THSZBR7:
(05296) /
(05297) / THSZBR8:
(05298) /
(05299) /
(05300) / DO S BUFFER...
(05301) / THSZBR8:
(05302) / THSZBR9:
(05303) / THSZBR10:

```

```

PAGE 119: (COND)(OCAL16>BMM16M.WSO.2, 29-Dec-88 14:30:40, Ed: VOLV
          APS) - THIST(V,W,V,W,R,S) APS PROGRAM FOR THIST

```

A0E	06EC6	1C020000	TMSS\$S11:	(05304)	ADD(R#0,M\$S)	J5 INCR (WHEN BOUND)
				(05305)		
				(05306)		
A0F	06EC8	1E0A0000	TMSS\$S12:	(05307)	ADD(R#0,M\$S,TF)	J5EN 3 ADORS (WHEN BOUND)
A10	06ECA	20190701	03	(05308)	SUBL(R#1,1),JUNPP(03)	JLOOP S0S TIMES
				(05309)		
A11	06EC0	22200031		(05310)	CLEAR(R#1)	DOONE WITH INPUT
A12	06EC0	24000020		(05311)	HOP(0)	
				(05312)		
				(05313)		
				(05314)	OUTPUT PROGRAM	
				(05315)	DO0: SAMPLE ADDRESSES	
				(05316)	001: BUFFER SIZES	
				(05317)		
A13	06ED0	26300032	TMSS\$0:	SET(R#1)		
				(05318)		
				(05319)		
				(05320)	DO Y BUFFER...	
A14	06ED2	28400000	TMSS\$Y0A:	LOAD(R#0,M\$S)	JY BAZE (WHEN BOUND)	
A15	06ED4	2A500000	TMSS\$0S2:	LOAD(R#1,M\$S)	J0 SIZE-1 (WHEN BOUND)	
A16	06ED6	2C020000	TMSS\$S11:	SUB(R#0,M\$S)	JY INCR (WHEN BOUND)	
				(05324)		
				(05325)		
A17	06ED8	2F0A0000	TMSS\$Y12:	04	ADD(R#0,M\$S,TF)	J5EN Y ADORS (WHEN BOUND)
A18	06EDA	30111701	04	(05327)	SUBL(R#1,1),JUNPP(04)	JLOOP 00S TIMES
				(05328)		
				(05329)	DO Y BUFFER...	
A19	06EDC	32400000	TMSS\$Y0A:	LOAD(R#0,M\$S)	JY BAZE (WHEN BOUND)	
A1A	06EDE	34500000	TMSS\$0S2:	LOAD(R#1,M\$S)	JM SIZE-1 (WHEN BOUND)	
A1B	06EDF	36020000	TMSS\$Y11:	SUB(R#0,M\$S)	JY INCR (WHEN BOUND)	
				(05333)		
				(05334)		
A1C	06EE2	300A0000	TMSS\$Y12:	05	ADD(R#0,M\$S,TF)	J5EN Y ADORS (WHEN BOUND)
A1D	06EE4	3A111C01	05	(05336)	SUBL(R#1,1),JUNPP(05)	JLOOP 00S TIMES
				(05337)		
				(05338)	DO R BUFFER...	
A1E	06EE6	3C400000	TMSS\$Y0A:	LOAD(R#0,M\$S)	JR BAZE (WHEN BOUND)	
A1F	06EE8	3E500000	TMSS\$0S2:	LOAD(R#1,M\$S)	J5 SIZE-1 (WHEN BOUND)	
A20	06EEA	40020000	TMSS\$S11:	SUB(R#0,M\$S)	JR INCR (WHEN BOUND)	
				(05342)		
				(05343)		
A21	06EEC	420A0000	TMSS\$S12:	06	ADD(R#0,M\$S,TF)	J5EN R ADORS (WHEN BOUND)
A22	06EE0	4412101	06	(05345)	SUBL(R#1,1),JUNPP(06)	JLOOP S0S TIMES
				(05346)		
				(05347)		
A23	06EF0	46200030		(05348)	CLEAR(R#0)	DOONE WITH OUTPUT
A24	06EF2	40000020		(05349)	HOP(0)	
				(05350)	TMSSC=0C	
				(05351)	END	
06EF4				(05352)		
				(05353)	TMSS2=0L-TMSS	

(05354) - SM5TH SPECIAL BINDING MODULE FOR TWIST
 (05355) ;
 (05356) ; PCD FORMAT (16 BIT FORMAT SHOWN)

```

WORD-----LEFT BYTE-----RIGHT BYTE-----
01      PTR TO NEXT PCB AND FUNCTION LIST FLAG (LSB)
10      CONTROL BITS
11      201

```

PTR TO NEXT PCB AND FUNCTION LIST FLAG (LSD)
CONTROL BITS
261

2017-2018

(05360)	30M\$TH1	PUSHMIL R3,APDT\$ONG(R2)	STACK APU MODULE BUS ORIGIN
(05369)	06B74	MOVIR R7,-H5	
(05370)	06B76	PUSHMIL R3,0-W5(R3)	STACK APU MODULE ST ADDR AND SZ
(05371)	06B78	PUSHMIL R3,APDT\$ONG+W5(R2)	STACK APS MODULE BUS ORIGIN
(05372)	06B7A	PUSHMIL R3,APDT\$ONG+2*W5(R2)	STACK CSPU SUPPORT MODULE
(05373)	06B7C	DECR R7,1	
(05374)	06B7E	REVEN	
(05375)	06B7F	PUSHMIL R3,0-2*W5(R3)	APS MODULE SIZE
(05376)	06B80	MOVIR R4,-3*W5(R3)	GET SPECIAL SUPPORT SEMAPHORE
(05377)	06B82	MOVIRE R4,R4,MSK\$BAYT	
(05378)	06B84	LRS R4,8	
(05379)	06B86	PUSHMIL R3,R4	STACK SPECIAL SUPPORT SEMAPHORE
(05380)	06B87	MOVIR R5,R1	POINT TO FIRST SCALAR ID
(05381)	06B88	INCR R5,H5	
(05382)	06B89	MOVIRE R4,R5,MSK\$BAYT	EXTRACT SCALAR A IDENTIFIER
(05383)	06B8A	PUSHMIL R3,R4	STACK SCALAR A IDENTIFIER
(05384)	06B8C	PUSHMIL R3,CLR\$CGCI	RESET 6 FLAGS
(05385)	06B8E	INCR R3,2	
(05386)	06B8F	PUSHMIL R3,ZERO	SET FOR NO PBY
(05387)	06B90	SUBIR R3,AP\$LLC	
(05388)	06B92		
(05389)	06B94		
(05390)	06B96		
(05391)	06B98		
(05392)	06B9A		
(05393)	06B9C		
(05394)	06B9E		
(05395)	06B9F		
(05396)	06BA0		
(05397)	06BA2		
(05398)	06BA4		
(05399)	06BA6		
(05400)	06BA8		
(05401)	06BA9		
(05402)	06BAE		
(05403)	06BAC		
(05404)	06BAF		
(05405)	06BB0		
(05406)	06BB2		
(05407)	06BB4		
(05408)	06BB6		
(05409)	06BB8		
(05410)	06BB9		
(05411)	06BBB		
(05412)	06BBE		
(05413)	06BBF		
(05414)	06BC0		
(05415)	06BC2		
(05416)	06BC4		
(05417)	06BC6		
(05418)	06BC8		
(05419)	06BCA		
(05420)	06BCB		
(05421)	06BCE		
(05422)	06BCF		
(05423)	06BD0		
(05424)	06BD2		
(05425)	06BD4		
(05426)	06BD6		
(05427)	06BD8		
(05428)	06BDA		
(05429)	06BDB		
(05430)	06BDD		
(05431)	06BDE		
(05432)	06BDF		
(05433)	06BD0		
(05434)	06BD2		
(05435)	06BD4		
(05436)	06BD6		
(05437)	06BD8		
(05438)	06BDA		
(05439)	06BDB		
(05440)	06BDD		
(05441)	06BDE		
(05442)	06BDF		
(05443)	06BD0		
(05444)	06BD2		
(05445)	06BD4		
(05446)	06BD6		
(05447)	06BD8		
(05448)	06BDA		
(05449)	06BDB		
(05450)	06BDD		
(05451)	06BDE		
(05452)	06BDF		
(05453)	06BD0		
(05454)	06BD2		
(05455)	06BD4		
(05456)	06BD6		
(05457)	06BD8		
(05458)	06BDA		
(05459)	06BDB		
(05460)	06BDD		
(05461)	06BDE		
(05462)	06BDF		
(05463)	06BD0		
(05464)	06BD2		
(05465)	06BD4		
(05466)	06BD6		
(05467)			

```

(05390) )
(05391) ) V UPPER BINDING:
(05392) )
(05393) ) V BASE....
(05394) ) MOVNR R6,NS(R1)
(05395) ) MOVNR R6,R6,MSEX$LEFT JGET BIO
(05396) ) LMS R6,7 JCONVERT TO DCT INDEX
(05397) ) EVFM
(05398) ) MOVNPL R4,DCT$R8(R6) JIDIT LEN., 2BIT BUSB, 17BIT BA.
(05399) ) MOVNR R7,THSS$N$*THSS$BA JGET LEFT NWORD OF INSTA
(05400) ) TORNR R4,R7,$FF$0 J"OR" IN NICH 4 BITS
(05401) ) MOVNML R4,THSS$N$*THSS$BA JMOVE PCWORD TO BUSI
(05402) )
(05403) ) VSI...
(05404) ) MOVNR R4,DCT$AD(R6) JSAMPLE INCREMENT
(05405) ) SAP R4,THSS$N$*THSS$YS11
(05406) ) SAP R4,THSS$N$*THSS$YS12

```

```

(05407) )
(05408) ) 0 BUFFER BINDING:
(05409) )
(05410) ) VBA...
MOVNR R6,MS(R1)
MOVNR R6,R6,MSKSRBYT JGET BID
LLS R6,1 JCONVERT TO BCT INDEX
EVEN
MOVNR R4,BCT$BA(R6) J10IT LEM., 20IT BUS8, 170IT BA.
MOVNR R7,THS$+MS*THSSUBA JGET LEFT WORD OF INSTR
TORNR R4,R7,$PFF8 J"OR" IN HIGH 4 BITS
MOVNR R4,THS$+MS*THSSUBA JMOVE FWORD TO BUS1
(05411) )
(05412) ) VBS...
MOVNR R4,MS+BC$AD(R6) JSIZE-1
MOVNR R4,THS$+MS*THSSUBS1+MS
MOVNR R4,THS$+MS*THSSUBS2+MS
(05413) )
(05414) ) VSI...
MOVNR R4,BCT$AD(R6) JSAMPLE INCREMENT
SAP
SAP
(05415) )
(05416) ) V BUFFER BINDING:
(05417) )
(05418) ) V BASE...
MOVNR R6,2*MS(R1)
MOVNR R6,R6,MSK$CMT JGET BID
LLS R6,7 JCONVERT TO BCT INDEX
EVEN
MOVNR R4,BCT$BA(R6) J10IT LEM., 20IT BUS8, 170IT BA.
MOVNR R7,THS$+MS*THSSUBA JGET LEFT WORD OF INSTR
TORNR R4,R7,$PFF8 J"OR" IN HIGH 4 BITS
MOVNR R4,THS$+MS*THSSUBA JMOVE FWORD TO BUS1
(05419) )
(05420) ) VSI...
MOVNR R4,BCT$AD(R6) JSAMPLE INCREMENT
SAP
SAP
(05421) )
(05422) ) V BUFFER BINDING:
(05423) )
(05424) ) V BASE...
MOVNR R6,2*MS(R1)
MOVNR R6,R6,MSK$CMT JGET BID
LLS R6,7 JCONVERT TO BCT INDEX
EVEN
MOVNR R4,BCT$BA(R6) J10IT LEM., 20IT BUS8, 170IT BA.
MOVNR R7,THS$+MS*THSSUBA JGET LEFT WORD OF INSTR
TORNR R4,R7,$PFF8 J"OR" IN HIGH 4 BITS
MOVNR R4,THS$+MS*THSSUBA JMOVE FWORD TO BUS1
(05425) )
(05426) ) VSI...
MOVNR R4,BCT$AD(R6) JSAMPLE INCREMENT
SAP
SAP
(05427) )
(05428) ) V BUFFER BINDING:
(05429) )
(05430) ) V BASE...
MOVNR R6,2*MS(R1)
MOVNR R6,R6,MSK$CMT JGET BID
LLS R6,7 JCONVERT TO BCT INDEX
EVEN
MOVNR R4,BCT$BA(R6) J10IT LEM., 20IT BUS8, 170IT BA.
MOVNR R7,THS$+MS*THSSUBA JGET LEFT WORD OF INSTR
TORNR R4,R7,$PFF8 J"OR" IN HIGH 4 BITS
MOVNR R4,THS$+MS*THSSUBA JMOVE FWORD TO BUS1
(05431) )
(05432) ) VSI...
MOVNR R4,BCT$AD(R6) JSAMPLE INCREMENT
SAP
SAP
(05433) )
(05434) ) V BUFFER BINDING:
(05435) )
(05436) ) V BASE...
MOVNR R6,2*MS(R1)
MOVNR R6,R6,MSK$CMT JGET BID
LLS R6,7 JCONVERT TO BCT INDEX
EVEN
MOVNR R4,BCT$BA(R6) J10IT LEM., 20IT BUS8, 170IT BA.
MOVNR R7,THS$+MS*THSSUBA JGET LEFT WORD OF INSTR
TORNR R4,R7,$PFF8 J"OR" IN HIGH 4 BITS
MOVNR R4,THS$+MS*THSSUBA JMOVE FWORD TO BUS1
(05437) )
(05438) ) VSI...
MOVNR R4,BCT$AD(R6) JSAMPLE INCREMENT
SAP
SAP
(05439) )
(05440) ) V BUFFER BINDING:
(05441) )
(05442) ) V BASE...
MOVNR R6,2*MS(R1)
MOVNR R6,R6,MSK$CMT JGET BID
LLS R6,7 JCONVERT TO BCT INDEX
EVEN
MOVNR R4,BCT$BA(R6) J10IT LEM., 20IT BUS8, 170IT BA.
MOVNR R7,THS$+MS*THSSUBA JGET LEFT WORD OF INSTR
TORNR R4,R7,$PFF8 J"OR" IN HIGH 4 BITS
MOVNR R4,THS$+MS*THSSUBA JMOVE FWORD TO BUS1
(05443) )
(05444) ) VSI...
MOVNR R4,BCT$AD(R6) JSAMPLE INCREMENT
SAP
SAP
(05445) )
(05446) ) V BUFFER BINDING:
(05447) )
(05448) ) V BASE...
MOVNR R6,2*MS(R1)
MOVNR R6,R6,MSK$CMT JGET BID
LLS R6,7 JCONVERT TO BCT INDEX
EVEN
MOVNR R4,BCT$BA(R6) J10IT LEM., 20IT BUS8, 170IT BA.
MOVNR R7,THS$+MS*THSSUBA JGET LEFT WORD OF INSTR
TORNR R4,R7,$PFF8 J"OR" IN HIGH 4 BITS
MOVNR R4,THS$+MS*THSSUBA JMOVE FWORD TO BUS1
(05449) )
(05450) ) VSI...
MOVNR R4,BCT$AD(R6) JSAMPLE INCREMENT
SAP
SAP
(05451) )
(05452) ) V BUFFER BINDING:
(05453) )
(05454) ) V BASE...
MOVNR R6,2*MS(R1)
MOVNR R6,R6,MSK$CMT JGET BID
LLS R6,7 JCONVERT TO BCT INDEX
EVEN
MOVNR R4,BCT$BA(R6) J10IT LEM., 20IT BUS8, 170IT BA.
MOVNR R7,THS$+MS*THSSUBA JGET LEFT WORD OF INSTR
TORNR R4,R7,$PFF8 J"OR" IN HIGH 4 BITS
MOVNR R4,THS$+MS*THSSUBA JMOVE FWORD TO BUS1
(05455) )
(05456) ) VSI...
MOVNR R4,BCT$AD(R6) JSAMPLE INCREMENT
SAP
SAP
(05457) )
(05458) ) V BUFFER BINDING:
(05459) )
(05460) ) V BASE...
MOVNR R6,2*MS(R1)
MOVNR R6,R6,MSK$CMT JGET BID
LLS R6,7 JCONVERT TO BCT INDEX
EVEN
MOVNR R4,BCT$BA(R6) J10IT LEM., 20IT BUS8, 170IT BA.
MOVNR R7,THS$+MS*THSSUBA JGET LEFT WORD OF INSTR
TORNR R4,R7,$PFF8 J"OR" IN HIGH 4 BITS
MOVNR R4,THS$+MS*THSSUBA JMOVE FWORD TO BUS1

```

PAGE 122: (000010C010)00010M.H00.2, 29-000-00 14:30:40, EDI WULF
S0M5TH SPECIAL BINDING MODULE FOR THIST

```

00764 F04C0605 (05460)      MOVNR R4,HS+BC$AD(R6)      )SIZE-1
00766 E0406E0B (05461)      MOVNR R4,TH$+W$-TH$S0S1+H$
00768 E0406E0F (05462)      MOVNR R4,TH$+W$-TH$S0S2+H$
                                )
                                )
0076A F04C0604 (05463)      )SI...
0076C E0406E0C (05464)      MOVNR R4,BC$AD(R6)      )SAMPLE INCREMENT
0076E E0406E0E (05465)      SAV R4,TH$+W$-TH$S0S1
00770 E0406E10 (05466)      SAV R4,TH$+W$-TH$S0S12
                                )
                                )
00772 F0620003 (05467)      ) R BUFFER BINDING:
00774 3C67      (05468)      )
00776 C04C0502 (05469)      ) R BASE...
00778 F0706E06 (05470)      MOVNR R6,3*W$(R1)
0077A 544F7F08 (05471)      MOVNR R6,R6,MSK$B0YT )GET BID
0077C E0406E0E (05472)      LRS R6,7          )CONVERT TO BCT INDEX
0077E F0706E06 (05473)      EVEN
00780 E0406E0E (05474)      MOVNR R4,BC$0A(R6) )BIT LEN,, 2BIT 0000, 17BIT 0A-
00782 E0406E0E (05475)      MOVNR R7,TH$+W$-TH$S0A )GET LEFT WORD OF INSTR
00784 E0406E0E (05476)      I0NR R4,R7,$FF00 )OR- IN HIGH 4 BITS
00786 E0406E0E (05477)      MOVNR R4,TH$+W$-TH$S0A )MOVE FWORD TO B0S1
00788 E0406E0E (05478)      )
0078A E0406E0E (05479)      )
0078C E0406E0E (05480)      )
                                )
0078E F04C0604 (05481)      ) SI...
00790 E0406E0A (05482)      MOVNR R4,BC$AD(R6)      )SAMPLE INCREMENT
00792 E0406E0A (05483)      SAV R4,TH$+W$-TH$S0S1
00794 E0406E0C (05484)      SAV R4,TH$+W$-TH$S0S12
                                )
                                )
00796 F0620003 (05485)      ) S BUFFER BINDING:
00798 3C61      (05486)      )
0079A C04C0502 (05487)      ) S0A...
0079C F0706E02 (05488)      MOVNR R6,3*W$(R1)
0079E F0706E02 (05489)      MOVNR R6,R6,MSK$B0YT )GET BID
007A0 E0406E02 (05490)      LLS R6,1          )CONVERT TO BCT INDEX
007A2 E0406E02 (05491)      EVEN
007A4 E0406E02 (05492)      MOVNR R4,BC$0A(R6) )BIT LEN,, 2BIT 0000, 17BIT 0A-
007A6 E0406E02 (05493)      MOVNR R7,TH$+W$-TH$S0A )GET LEFT WORD OF INSTR
007A8 E0406E02 (05494)      I0NR R4,R7,$FF00 )OR- IN HIGH 4 BITS
007AA E0406E02 (05495)      MOVNR R4,TH$+W$-TH$S0A )MOVE FWORD TO B0S1
007AC E0406E02 (05496)      )
007AE E0406E02 (05497)      )
                                )
007B0 F04C0605 (05498)      ) S0S...
007B2 E0406E05 (05499)      MOVNR R4,HS+BC$AD(R6)      )SIZE-1
007B4 E0406E05 (05500)      MOVNR R4,TH$+W$-TH$S0S1+H$
007B6 E0406E05 (05501)      MOVNR R4,TH$+W$-TH$S0S2+H$
                                )
                                )
007B8 F04C0604 (05502)      ) SI...
007BA E0406E04 (05503)      MOVNR R4,BC$AD(R6)      )SAMPLE INCREMENT
007BC E0406E04 (05504)      SAV R4,TH$+W$-TH$S0S1
007BE E0406E04 (05505)      SAV R4,TH$+W$-TH$S0S12
                                )
                                )
007C0 E0000F03 (05506)      )
007C2 E0000F03 (05507)      )
007C4 E0000F03 (05508)      )
007C6 E0000F03 (05509)      JMP IP$B0R00

```



```

(05592) * APS3 - SCLRES(V,A,U,V)
(05593) * WNR 9/80
(05594) *
(05595) * APS ROUTINE FOR SCL
(05596) *
(05597) * INPUTS:
(05598) * IQ: SA,U(1),U(2),U(3),V(1),...,V(216)
(05599) * OUTPUT:
(05600) * OQ: DUM,DUM,V(1),...,V(216)
(05601) *
(05602) * INPUT PROGRAM REGISTER USAGE
(05603) * BR0 SA,U,V
(05604) * BR1 SIZE (UNUSED)
(05605) * BR2 COUNTER(72)
(05606) * BR3 COUNTER(3)
(05607) *
(05608) * HEADER BLOCK
(05609) * EVEN
(05610) * ADDR SCL$1
(05611) * ADDR SCL$+2*SCL$5
(05612) * DATA 1
(05613) * DATA SCL$2
(05614) * ADDR SCL$3A
(05615) * EVEN
(05616) *
(05617) * END OF HEADER BLOCK
(05618) *
(05619) * START OF APS PROGRAM
(05620) SCL$1: BEGIN APS(SCL$)
(05621) *
(05622) * JSM(SCL$OP,P2)
(05623) * SET(MO)
(05624) * INPUT SCALER
(05625) SCL$5: LOAD(BR0,M$$(1),TF)
(05626) *
(05627) * LOAD DELTA GAINS - U
(05628) * LOAD(BR0,C13,TF)
(05629) * LOAD(BR1,M$$(1))
(05630) * ADD(BR0,M$$(1),TF)
(05631) * ADD(BR0,C13,TF)
(05632) *
(05633) * LOAD RESIDUAL SAMPLES
(05634) * LOAD(BR0,C23)
(05635) * LOAD(BR1,M$$(1))
(05636) * SUB(BR0,M$$(1))
(05637) * LOAD COUNTERS IN BR2 AND BR3
(05638) * LOAD(BR3,-1)
(05639) SCL$5: LOAD(BR2,72-2)
(05640) * LOOP FOR SEGMENT SAMPLES INPUT
(05641) SCL$2: ADD(BR0,C103,TF)
(05642) * SUBL(BR2,1),JUMPP(SCL$52)
(05643) * ADDL(BR0,M$$(1),JUMPI(BA+1))
(05644) * SET(VI)

```

)PTR TO CONSTRUCTED INSTRUCTION BLOCK
)PTR TO SCALER BLOCK
)SET NUMBER OF SCALERS
)SET MODULE SIZE
)POINTER TO CHAIN ANCHOR
)START APS PRC ON FULL WORDS

)LOAD AND GEN SA ADDR(GAIN)

)LOAD AND GEN U(1) ADDR
)GEN U(2) ADDR
)GEN U(3) ADDR

)LOAD V SA
)GEN U(1) ADDR
)PREPARE TO INC LATER

)BR3=8 OF SEGMENTS - 1
)BR2=8 OF SAMPLES/SEC - 2

)GEN V(1) ADDR
)DEC INCR AND JUMP
)ON LAST VALUE AND ENSURE SET VI
)SIGNAL END OF INPUT

PAGE 126: [BUND]C[AL]6[BU]16[M.NSD.2, 29-Dec-88 14:38:48, ED: VOLF
AP33 - SCLAES(Y,A,U,V)

266

PAGE 120: (0000) <0C116>00016M.N50.2, 29-Dec-88 14:30:40, Ed: WOLF
AP03-PITSVM(V,A,U,V) 3-TAP PITCH SYNTHESIS FILTER

```

0702C 0000 (05730) DATA PITSUSSA JAPU START ADDRESS
0702D 0020 (05739) DATA PITSUSZ JSIZE
(05740)
(05741) PITSUSZ: BEGIN APU(PITSVM)
000 0702E 00E00000 (05742) MOV(IQA,M0) \ NOP
001 07030 00E00000 (05743) MOV(IQA,M4) \ NOP
002 07032 04000000 (05744) MUL(M0,M4) \ NOP
003 07034 00000000 (05745) MOV(P,A0) \ NOP
004 07036 31000000 (05746) ALIGN(A0) \ NOP
005 07038 009C0000 (05747) MOV(R,OQ) \ NOP
(05748) HERE WE WAIT ON OQE, PROCEED APS, READ/WRITE THE INSTRUCTION, WAIT
(05749) AND PROCEED THE APS AGAIN, AND THEN GET DOWN TO THE COMPUTATION.
006 0703A 90100000 (05750) B1: JUMPC(R1,OQE)
007 0703C 00000000 (05751) NOP
008 0703E 20372037 (05752) CLEAR(M1)
009 07040 00PC0000 (05753) MOV(IQA,OQ) \ NOP
00A 07042 90100000 (05754) B2: JUMPC(R2,OQE)
00B 07044 00000000 (05755) NOP
00C 07046 20372037 (05756) CLEAR(M1)
00D 07048 00EC00EE (05757) MOV(IQA,M4) \ MOV(IQA,M6)
00E 0704A 00E000EC (05758) MOV(IQA,M5) \ MOV(IQA,M4)
00F 0704C 00E000ED (05759) MOV(IQA,M6) \ MOV(IQA,M5)
(05760) MAIN LOOP
010 0704E 00E000E0 (05761) B3: MOV(IQA,M0)
011 07050 04104111 (05762) MOV(A1),MUL(M0,M4)
012 07052 49104910 (05763) MOV(A0),SUB(A0,A1)
013 07054 00F20000 (05764) MOV(IQA,A2) \ NOP
014 07056 000000F2 (05765) NOP \ MOV(IQA,A2)
015 07058 009C009C (05766) MOV(R,OQ)
016 0705A 00E900E9 (05767) MOV(IQA,M1)
017 0705C 04010401 (05768) MOV(A1),MUL(M1,M5)
018 0705E 49404940 (05769) SUB(A2,A1)
019 07060 00E000E0 (05770) MOV(IQA,M2) \ NOP
01A 07062 000000EA (05771) NOP \ MOV(IQA,M2)
01B 07064 05510551 (05772) MOV(A1),MUL(M2,M6)
01C 07066 49104910 (05773) MOV(A0),SUB(A0,A1)
01D 07068 901C0010 (05774) JUMPC(R3,ED)
01E 0706A 20372032 (05775) CLEAR(M4)
01F 0706C 10000000 (05776) JUMP(R0)
(05777) PITSUSZ:=PA-PITSUSSA
0706E 00000020 (05778) END

```

(05779) - APS3-PITSYN(V,A,U,V) APS PROGRAM FOR PITSYN

```

(05783) ) THIS APS PROGRAM MODIFIES ONE INSTRUCTION IN ITSELF BY:
(05784) ) 0. READING PITCH FROM SCALAR A, AND FIXING IT
(05785) ) 1. WRITING IT INTO THE RM OF THE BUS-1 COPY OF THE INSTRUCTION
(05786) ) 2. WAITING ON ONE TO BE SURE IT'S THERE (PLUS 1 NOP THAT STORES
(05787) )   SAYS IS NECESSARY)
(05788) ) 3. READING THE MODIFIED APS INSTRUCTION (FULLWORD) FROM BUS 1
(05789) ) 4. WRITING IT TO APSMEN IN PSEUDOMEMORY
(05790) ) 5. WAITING FOR ONE AGAIN BEFORE LETTING THE APS INPUT PROGRAM
(05791) )   PROCEED.

```

(05792) ,
(05793) ,
(05794) ,

[illegible]

```

(05067) ) INPUT PROGRAM
(05068) ) DOB: V(2), THEN V(N-N+1)
(05069) ) DO(1) U(N)
(05070) ) DO(2) LOOP COUNTER, START
(05071) )

```

```

(05000) PITSS1: BEGIN APS(PITSS)
(05009) JSM(PITSS0,P2)
(05010) SET(RO)
(05011) LOAD(ORG,SVTSM(1),TP)
(05012) PITSS$: LOAD(ORG,M$(1),TP)
(05013) SET(WI)
(05014) WOP(0)
(05015) LOAD(ORG,PITSSM(1),TP)
(05016) SET(WI)
)SET AND START OUTPUT PROGRAM
)GEN (2*-15) ARE
)GEN SA ADDRESS (PITCH)
)WAIT FOR APU TO SIGNAL WRITE COMPLETE
)LOAD MODIFIED INSTR THRU APU INTO APS
)WAIT AGAIN FOR WRITE TO COMPLETE

```

```

(05017)      MOV     R0,R0
(05018)      LOAD(R0,C23,TF)
(05019)      LOAD(R0,M$S)
(05020)      ADD(R0,M$S,TF)
(05021)      ADC(R0,C10,TF)
(05022)      NOW    LOAD INFO FOR ACCESSING THE Y AND W BUFFERS
(05023)      ,YBA
(05024)      LOAD(R0,C8A)
(05025)      LOAD(R0,M$S)
(05026)      ADD(R0,C216*W$)
(05027)      NEXT INSTRUCTION GETS THE PITCH JAMMED INTO IT!
          OFFSET FOR 216 PA
          (UNUSED) Y INCR
          (UNUSED) Y INCR
          (UNUSED) Y LGTM-1
          (UNUSED) V LGTM-1
          GEN V(1)
          GEN V(2)
          GEN V(0)

```

```

(05020) PIT3$M=BL
          LOAD(002,M$)
(05020)          ADD0(002,002)
(05031)          SUB0(000,002)
          POINT TO Y(0-M)
          J2-PITCH FOR FULLWORD ADDRESSING
          J2-PITCH
(05020)

```

```

PAGE 1301 (0000) (DCAL16) DBM16M.MSO.2, 29-Dec-88 14:30:40, Ed: WOLF
          APS3-PITSSN(V,A,W,V) APS PROGRAM FOR PITSSN

A14 0709E 2050100E (05032) LOAD(BR1,C13)
A15 0709E 20600000 (05033) LOAD(BR2,M$)
A16 0709E 20120000 (05034) SUB(BR1,M$)
A17 0709E 2009003A (05035) JMAIN LOOP
A18 0709E 30919000 (05036) #4: ADDL(BR0,M$,TF)
A19 0709E 30919000 (05037) ADD(BR1,C9,TF)
A19 0709E 32919002 (05038) ADD(BR1,C9,TF)
A1A 0709E 34090032 (05039) SUBL(BR0,M$,TF)
A1B 0709E 36090032 (05040) SUBL(BR0,M$,TF)
A1C 0709E 38090032 (05041) ADDL(BR0,M$,TF)
A1D 0709E 3A291702 (05042) SUML(BR2,2),JUNPP(04)
A1E 0709E 3C200031 (05043) CLEAR(R1)
A1F 0709E 3E000020 (05044) NOP(0)
A1F 0709E 3E000020 (05045) J
A1F 0709E 3E000020 (05046) J OUTPUT PROGRAM
A1F 0709E 3E000020 (05047) J BWP: V(M)
A1F 0709E 3E000020 (05048) J BWP: LOOP COUNTER (STARTS AT BWS-1)
A1F 0709E 3E000020 (05049) J BWP: SCRATCH
A1F 0709E 3E000020 (05050) J
A20 0709E 40300032 (05051) PITSS0: SET(RA)
A21 0709E 43727099 (05052) JGENERATE BUS1 ADR FOR THE INSTRUCTION TO BE MODIFIED (RIGHT HALF)
A21 0709E 43727099 (05053) LOAD(BW3,PITSSM+M$(1),TF)
A22 0709E 44F37FC0 (05054) JGENERATE PSEUDOMEMORY ADR FOR LOADING THE APS MEMORY
A22 0709E 44F37FC0 (05055) LOAD(BW3,APSMEH(1),TF)
A22 0709E 44F37FC0 (05056) JNOW GET ON WITH THE "ORDINARY" OUTPUT
A23 0709E 46F20794 (05057) LOAD(BW3,DW$(1),TF)
A24 0709E 48F20794 (05058) LOAD(BW3,DW$(1),TF)
A25 0709E 4A400010 (05059) LOAD(BW0,C03)
A26 0709E 4C700000 (05060) LOAD(BW3,M$)
A27 0709E 4E020000 (05061) SUB(BW0,M$)
A28 0709E 50001000 (05062) ADD(BW0,216*W$)
A29 0709E 52701000 (05063) LOAD(BW3,C13)
A2A 0709E 54500000 (05064) LOAD(BW1,M$)
A2B 0709E 56320000 (05065) SUB(BW3,M$)
A2B 0709E 56320000 (05066) J OUTPUT LOOP
A2C 0709E 5801003A (05067) #5: ADDL(BW0,M$,TF)
A2D 0709E 5A112CB1 (05068) SUBL(BW1,1),JUNPP(05)
A2E 0709E 5C200030 (05069) CLEAR(R0)
A2F 0709E 5E000020 (05070) NOP(0)
A2F 0709E 5E000020 (05071) END
A2F 0709E 5E000020 (05072) PITSS0: EC
A2F 0709E 5E000020 (05073) PITSS0: DATA OF "0"
...
A2F 0709E 5E000020 (05074) PITSS0: #L-PITSS0
          JASSIGN VALUE TO CHAIN ANCHOR
          JBLOCK FOR CONSTRUCTED INSTRUCTIONS
          JCOMPUTE BUS1 MODULE SIZE

```

PAGE 131: (0000)<0CAL0>00010M-MSU.2, 29-00C-00 14:30:40, E01 VOLV
APU3 - DEAL(V,A,W,B,V,W)

```
(05075) * APU3 - DEAL(V,A,W,B,V,W)
(05076) * FIELD 9/80
(05077) *
(05078) * RUNS WITH DEAL$S
(05079) *
(05080) * FUNCTION:
(05081) * DEAL DECODED RECEIVED DATA FROM INPUT BUFFER V.
(05082) * V ->
(05083) * ( 1) DECODED APC RESIDUAL SAMPLES
(05084) * ( 3) DECODED ENERGY
(05085) * ( 3) DECODED DELTA GAINS
(05086) * ( 1) DECODED PITCH LAG
(05087) * ( 3) DECODED PITCH PREDICTOR COEFFS
(05088) * ( 6) DECODED SPECTRAL PREDICTOR COEFFS
(05089) * -----
(05090) * (230)
```

THE 216 DECODED APC RESIDUAL SAMPLES ARE NOT MOVED FROM THE V
BUFFER. THEY ARE SUBSEQUENTLY REFERENCED DIRECTLY AS V(0-215).
ALL PARAMETERS (BUFFERS & SCALARS) ARE TREATED AS BL,LNG

```
V(216) => SA
V(217-219) => U(0-2)
V(220) => SB
V(221-223) => V(0-2)
V(224-229) => W(0-5)
```

```
INPUT SEQUENCE:
V(216), ..., V(229), (PI)
```

```
OUTPUT SEQUENCE:
SA, U(0), U(1), U(2),
SB, V(0), V(1), V(3),
W(0), ..., W(5), (EO)
```

```
EVEN DEAL$SA
DATA DEAL$SZ
```

```
DEAL$S BEGIN APU(DEAL$)
DEAL$SA:
MOV(10A,00) \ NOP
JUMPC(DEAL$SA,EO)
```

*
;COPY INPUT TO OUTPUT
;LOOP TIL DONE

```
CLEAR(RA)
JUMP(0)
DEAL$SZ = RA-DEAL$SA
END
```

070E6 0000
070E7 0004

A00 070E8 00FC0000
A01 070FA 90C00000
A02 070FC 2032032
A03 070FE 10000000
00000004
070FF
05921

PAGE 133: (PROM) <OCAL>00V16M.W50.2, 29-Dec-88 14:38:48, Ed: VOLV
AP33 -- DEAL(T,A,B,B,V,W)

ADDRESS	OPERATION	COMMENT
05964	GEN W(0)	START PCN
05965	GEN W(0)	START PCN
05966	GEN W(0)	START PCN
05967	DEALS\$0	SET(0A)
05968	DEALS\$0	SET(0A)
05969	DEALS\$0	SET(0A)
05970	LOAD(RW0,M\$\$(1),TF)	LOAD(RW0,M\$\$(1),TF)
05971	LOAD(RW0,M\$\$(1))	LOAD(RW0,M\$\$(1))
05972	GEN W(0-2)	GEN W(0-2)
05973	GEN W(0-2)	GEN W(0-2)
05974	LOAD(RW1,C13,TF)	LOAD(RW1,C13,TF)
05975	LOAD(RW2,M\$\$(1))	LOAD(RW2,M\$\$(1))
05976	LOAD(RW2,M\$\$(1))	LOAD(RW2,M\$\$(1))
05977	LOAD(RW2,M\$\$(1))	LOAD(RW2,M\$\$(1))
05978	ADD(RW1,C13,TF)	ADD(RW1,C13,TF)
05979	ADD(RW1,C13,TF)	ADD(RW1,C13,TF)
05980	ADD(RW1,C13,TF)	ADD(RW1,C13,TF)
05981	GEN SB	GEN SB
05982	GEN SB	GEN SB
05983	GEN SB	GEN SB
05984	MOVE(RW0,RW0,TF)	MOVE(RW0,RW0,TF)
05985	GEN V(0-2)	GEN V(0-2)
05986	GEN V(0-2)	GEN V(0-2)
05987	GEN V(0-2)	GEN V(0-2)
05988	LOAD(RW1,C23,TF)	LOAD(RW1,C23,TF)
05989	LOAD(RW2,M\$\$(1))	LOAD(RW2,M\$\$(1))
05990	LOAD(RW2,M\$\$(1))	LOAD(RW2,M\$\$(1))
05991	LOAD(RW2,M\$\$(1))	LOAD(RW2,M\$\$(1))
05992	ADD(RW1,C103,TF)	ADD(RW1,C103,TF)
05993	ADD(RW1,C103,TF)	ADD(RW1,C103,TF)
05994	GEN W(0-5)	GEN W(0-5)
05995	GEN W(0-5)	GEN W(0-5)
05996	GEN W(0-5)	GEN W(0-5)
05997	LOAD(RW1,C33)	LOAD(RW1,C33)
05998	LOAD(RW2,M\$\$(1))	LOAD(RW2,M\$\$(1))
05999	SUB(RW1,M\$\$(1))	SUB(RW1,M\$\$(1))
06000	LOAD(RW2,C-1)	LOAD(RW2,C-1)
06001	LOAD(RW1,C13,TF)	LOAD(RW1,C13,TF)
06002	SUB(RW2,C13,TF)	SUB(RW2,C13,TF)
06003	GEN W(0-1)	GEN W(0-1)
06004	GEN W(0-1)	GEN W(0-1)
06005	GEN W(0-1)	GEN W(0-1)
06006	GEN W(0-1)	GEN W(0-1)
06007	GEN W(0-1)	GEN W(0-1)
06008	DEALS\$A-BC	DEALS\$A-BC
06009	DEALS\$A-BC	DEALS\$A-BC
06010	END	END
06011	END	END
06012	DEALS\$A DATA	DEALS\$A DATA
06013	DEALS\$A DATA	DEALS\$A DATA
06014	DEALS\$A DATA	DEALS\$A DATA
06015	DEALS\$A DATA	DEALS\$A DATA
06016	DEALS\$A DATA	DEALS\$A DATA
06017	DEALS\$A DATA	DEALS\$A DATA
06018	DEALS\$A DATA	DEALS\$A DATA
06019	DEALS\$A DATA	DEALS\$A DATA
06020	DEALS\$A DATA	DEALS\$A DATA
06021	DEALS\$A DATA	DEALS\$A DATA
06022	DEALS\$A DATA	DEALS\$A DATA
06023	DEALS\$A DATA	DEALS\$A DATA
06024	DEALS\$A DATA	DEALS\$A DATA
06025	DEALS\$A DATA	DEALS\$A DATA
06026	DEALS\$A DATA	DEALS\$A DATA
06027	DEALS\$A DATA	DEALS\$A DATA
06028	DEALS\$A DATA	DEALS\$A DATA
06029	DEALS\$A DATA	DEALS\$A DATA
06030	DEALS\$A DATA	DEALS\$A DATA
06031	DEALS\$A DATA	DEALS\$A DATA
06032	DEALS\$A DATA	DEALS\$A DATA
06033	DEALS\$A DATA	DEALS\$A DATA
06034	DEALS\$A DATA	DEALS\$A DATA
06035	DEALS\$A DATA	DEALS\$A DATA
06036	DEALS\$A DATA	DEALS\$A DATA
06037	DEALS\$A DATA	DEALS\$A DATA
06038	DEALS\$A DATA	DEALS\$A DATA
06039	DEALS\$A DATA	DEALS\$A DATA
06040	DEALS\$A DATA	DEALS\$A DATA
06041	DEALS\$A DATA	DEALS\$A DATA
06042	DEALS\$A DATA	DEALS\$A DATA
06043	DEALS\$A DATA	DEALS\$A DATA
06044	DEALS\$A DATA	DEALS\$A DATA
06045	DEALS\$A DATA	DEALS\$A DATA
06046	DEALS\$A DATA	DEALS\$A DATA
06047	DEALS\$A DATA	DEALS\$A DATA
06048	DEALS\$A DATA	DEALS\$A DATA
06049	DEALS\$A DATA	DEALS\$A DATA
06050	DEALS\$A DATA	DEALS\$A DATA
06051	DEALS\$A DATA	DEALS\$A DATA
06052	DEALS\$A DATA	DEALS\$A DATA
06053	DEALS\$A DATA	DEALS\$A DATA
06054	DEALS\$A DATA	DEALS\$A DATA
06055	DEALS\$A DATA	DEALS\$A DATA
06056	DEALS\$A DATA	DEALS\$A DATA
06057	DEALS\$A DATA	DEALS\$A DATA
06058	DEALS\$A DATA	DEALS\$A DATA
06059	DEALS\$A DATA	DEALS\$A DATA
06060	DEALS\$A DATA	DEALS\$A DATA
06061		

A0A 07164 04550435 (06060)	MOV(A5), MUL(M0, M6) \ MOV(A5), MUL(M0, M5)	A5=P01(K-1), P03(K-2) \ A5=P02(K-2), P0
A0B 07166 4534554 (06069)	MOV(A3), ADD(A1, A5) \ MOV(A4), ADD(A2, A5)	A3=P03(K-3), S1(K-1) \ A4=S4(K-4), S2(K-
A0C 07168 08C000C (06070)	MOV(IQ1, M4)	M4=V(K)
A0D 0716A 04160476 (06071)	MOV(A6), MUL(M0, M4) \ MOV(A6), MUL(M0, M7)	A6=P03(K-2), P01(K) \ A6=P04(K-3), P02(K
A0E 0716C 46714692 (06072)	MOV(A1), ADD(A3, A6) \ MOV(A2), ADD(A4, A6)	A1=S1(K-1), S3(K-2) \ A2=S2(K-2), S4(K-
A0F 0716E 91160026 (06073)	JUMPS(DCCL0, PVI)	EXIT IF K=M-1 (LAST REF SAMPLE INDE
A10 07170 00E900E9 (06074)	•	
A11 07172 04F50405 (06075)	DCOR1	
A12 07174 4534554 (06076)	MOV(IQ1, M1)	M1=V(K+1)
A13 07176 08E000E0 (06077)	MOV(A5), MUL(M1, M7) \ MOV(A5), MUL(M1, M6)	A5=P01(K), P03(K-1) \ A5=P02(K-1), P04(K
A14 07178 04060496 (06078)	MOV(A3), ADD(A1, A5) \ MOV(A4), ADD(A2, A5)	A3=P03(K-2), S1(K) \ A4=S4(K-3), S2(K-1)
A15 0717A 46714692 (06079)	MOV(IQ1, M5)	M5=V(K+1)
A16 0717C 91160030 (06080)	MOV(A6), MUL(M1, M5) \ MOV(A6), MUL(M1, M4)	A6=P03(K-1), P01(K+1) \ A6=P04(K-2), P0
A17 0717E 00E000E0 (06081)	MOV(A1), ADD(A3, A6) \ MOV(A2), ADD(A4, A6)	A1=S1(K), S3(K-1) \ A2=S2(K-1), S4(K-2)
A18 07180 04550435 (06082)	JUMPS(DCCL1, PVI)	EXIT IF K+1=M-1 (LAST REF SAMP INDE
A19 07182 04F50405 (06083)	•	
A20 07184 08E000E0 (06084)	DCOR2	
A21 07186 04550435 (06085)	MOV(IQ1, M0)	M0=V(K+2)
A22 07188 04F50406 (06086)	MOV(A5), MUL(M0, M4) \ MOV(A5), MUL(M0, M7)	A5=P01(K+1), P03(K) \ A5=P02(K), P04(K-
A23 0718A 46714692 (06087)	MOV(A3), ADD(A1, A5) \ MOV(A4), ADD(A2, A5)	A3=P03(K-1), S1(K+1) \ A4=S4(K-2), S2(K
A24 0718C 91160054 (06088)	MOV(IQ1, M6)	M6=V(K+2)
A25 0718E 00E900E9 (06089)	MOV(A6), MUL(M0, M6) \ MOV(A6), MUL(M0, M5)	A6=P03(K), P01(K+2) \ A6=P04(K-1), P02(K
A26 07190 04550435 (06090)	MOV(A1), ADD(A3, A6) \ MOV(A2), ADD(A4, A6)	A1=S1(K+1), S3(K) \ A2=S2(K), S4(K-1)
A27 07192 08E000E0 (06091)	JUMPS(DCCL2, PVI)	EXIT IF K+2=M-1 (LAST REF SAMP INDE
A28 07194 46714692 (06092)	•	
A29 07196 91160054 (06093)	DCOR3	
A30 07198 00E900E9 (06094)	MOV(IQ1, M1)	M1=V(K+3)
A31 0719A 04F50406 (06095)	MOV(A5), MUL(M1, M5) \ MOV(A5), MUL(M1, M4)	A5=P01(K+2), P03(K+1) \ A5=P02(K+1), P0
A32 0719C 4534554 (06096)	MOV(A3), ADD(A1, A5) \ MOV(A4), ADD(A2, A5)	A3=P03(K), S1(K+2) \ A4=S4(K-1), S2(K+1)
A33 0719E 08E000E0 (06097)	MOV(IQ1, M7)	M7=V(K+3)
A34 0719A 04F50406 (06098)	MOV(A6), MUL(M1, M7) \ MOV(A6), MUL(M1, M6)	A6=P03(K+1), P01(K+3) \ A6=P04(K), P02(K
A35 0719C 46714692 (06099)	MOV(A1), ADD(A3, A6) \ MOV(A2), ADD(A4, A6)	A1=S1(K+2), S3(K+1) \ A2=S2(K+1), S4(K)
A36 0719E 91160068 (06100)	JUMPS(DCCL3, PVI)	EXIT IF K+3=M-1 (LAST REF SAMP INDE
A37 0719A 10000000 (06101)	•	
A38 0719C 10000000 (06102)	JUMP(DCCL0)	
A39 0719E 20372037 (06103)	•	
A40 0719A 08E900E9 (06104)	ENTRY FOR K=M-1 CLEAN-UP	
A41 0719C 08E900E9 (06105)	CLEAR(M1)	
A42 0719E 04F50405 (06106)	MOV(IQ1, M1)	M1=V(M)
A43 0719A 04F50405 (06107)	MOV(A5), MUL(M1, M7) \ MOV(A5), MUL(M1, M6)	A5=P01(K-1), P03(M-2) \ A5=P02(M-2), P0
A44 0719C 4534554 (06108)	MOV(A3), ADD(A1, A5) \ MOV(A4), ADD(A2, A5)	A3=P03(M-3), S1(M-1) \ A4=S4(M-4), S2(M-
A45 0719E 04F50406 (06109)	MOV(A6), MUL(M1, M5) \ MOV(A6), MUL(M1, M4)	A6=P03(M-2), P01(M) \ A6=P04(M-3), P02(M
A46 0719A 46714692 (06110)	MOV(OQ), ADD(A3, A6) \ MOV(A2), ADD(A4, A6)	OQ=S1(M-1), S3(M-2) \ A2=S2(M-2), S4(M
A47 0719C 08E900E9 (06111)	MOV(ZERO, A1)	CLEAR S1 FOR NEW SUM
A48 0719A 08E900E9 (06112)	•	
A49 0719C 08E900E9 (06113)	MOV(IQ1, M0)	M0=V(M+1)
A50 0719E 04F50405 (06114)	MOV(A5), MUL(M0, M4) \ MOV(A5), MUL(M0, M7)	A5=P01(M), P03(M-1) \ A5=P02(M-1), P04(M
A51 0719A 4534554 (06115)	MOV(A3), ADD(A1, A5) \ MOV(A4), ADD(A2, A5)	A3=P03(M-2), S1(M) \ A4=S4(M-3), S2(M-1)
A52 0719C 04F50406 (06116)	MOV(A6), MUL(M0, M6) \ MOV(A6), MUL(M0, M5)	A6=P03(M-1), P01(M+1) \ A6=P04(M-2), P0
A53 0719E 91160082 (06117)	JUMPS(DCEND, EU)	JUMP TO END IF NO MORE OUTPUT
A54 0719A 46714692 (06118)	MOV(A1), ADD(A3, A6) \ MOV(OQ), ADD(A4, A6)	A1=S1(M), S3(M-1) \ OQ=S2(M-1), S4(M-2
A55 0719C 08E900E9 (06119)	MOV(ZERO, A2)	CLEAR S2 FOR NEW SUM
A56 0719A 08E900E9 (06120)	•	
A57 0719C 08E900E9 (06121)	MOV(IQ1, M1)	M1=V(M+2)
A58 0719A 04F50405 (06122)	MOV(A5), MUL(M1, M5) \ MOV(A5), MUL(M1, M4)	A5=P01(M+1), P03(M) \ A5=P02(M), P04(M-
A59 0719C 4534554 (06123)	JUMPS(DCEND, EU)	JUMP TO END IF NO MORE OUTPUT

```

A37 0710E 453C4554 (06121)
A38 0710C 08130013 (06122)
A39 0710C 04F60406 (06123)
A3A 0710C 46714692 (06124)
A3B 0710C 0800009C (06125)
A3C 0710C 10000000 (06126)
A3D 0710C 10000000 (06127)
A3E 0710C 20372037 (06128)
A3F 0710C 080000E0 (06129)
A40 0710C 04F60406 (06130)
A41 0710C 04F60406 (06131)
A42 0710C 04F60406 (06132)
A43 0710C 04F60406 (06133)
A44 0710C 04F60406 (06134)
A45 0710C 04F60406 (06135)
A46 0710C 04F60406 (06136)
A47 0710C 04F60406 (06137)
A48 0710C 04F60406 (06138)
A49 0710C 04F60406 (06139)
A4A 0710C 04F60406 (06140)
A4B 0710C 04F60406 (06141)
A4C 0710C 04F60406 (06142)
A4D 0710C 04F60406 (06143)
A4E 0710C 04F60406 (06144)
A4F 0710C 04F60406 (06145)
A50 0710C 04F60406 (06146)
A51 0710C 04F60406 (06147)
A52 0710C 04F60406 (06148)
A53 0710C 04F60406 (06149)
A54 0710C 04F60406 (06150)
A55 0710C 04F60406 (06151)
A56 0710C 04F60406 (06152)
A57 0710C 04F60406 (06153)
A58 0710C 04F60406 (06154)
A59 0710C 04F60406 (06155)
A5A 0710C 04F60406 (06156)
A5B 0710C 04F60406 (06157)
A5C 0710C 04F60406 (06158)
A5D 0710C 04F60406 (06159)
A5E 0710C 04F60406 (06160)
A5F 0710C 04F60406 (06161)
A60 0710C 04F60406 (06162)
A61 0710C 04F60406 (06163)
A62 0710C 04F60406 (06164)
A63 0710C 04F60406 (06165)
A64 0710C 04F60406 (06166)
A65 0710C 04F60406 (06167)
A66 0710C 04F60406 (06168)
A67 0710C 04F60406 (06169)
A68 0710C 04F60406 (06170)
A69 0710C 04F60406 (06171)
A70 0710C 04F60406 (06172)
A71 0710C 04F60406 (06173)
A72 0710C 04F60406 (06174)
A73 0710C 04F60406 (06175)
A74 0710C 04F60406 (06176)
A75 0710C 04F60406 (06177)
A76 0710C 04F60406 (06178)
A77 0710C 04F60406 (06179)
A78 0710C 04F60406 (06180)
A79 0710C 04F60406 (06181)
A80 0710C 04F60406 (06182)
A81 0710C 04F60406 (06183)
A82 0710C 04F60406 (06184)
A83 0710C 04F60406 (06185)
A84 0710C 04F60406 (06186)
A85 0710C 04F60406 (06187)
A86 0710C 04F60406 (06188)
A87 0710C 04F60406 (06189)
A88 0710C 04F60406 (06190)
A89 0710C 04F60406 (06191)
A90 0710C 04F60406 (06192)
A91 0710C 04F60406 (06193)
A92 0710C 04F60406 (06194)
A93 0710C 04F60406 (06195)
A94 0710C 04F60406 (06196)
A95 0710C 04F60406 (06197)
A96 0710C 04F60406 (06198)
A97 0710C 04F60406 (06199)
A98 0710C 04F60406 (06200)
A99 0710C 04F60406 (06201)
A9A 0710C 04F60406 (06202)
A9B 0710C 04F60406 (06203)
A9C 0710C 04F60406 (06204)
A9D 0710C 04F60406 (06205)
A9E 0710C 04F60406 (06206)
A9F 0710C 04F60406 (06207)
AA0 0710C 04F60406 (06208)
AA1 0710C 04F60406 (06209)
AA2 0710C 04F60406 (06210)
AA3 0710C 04F60406 (06211)
AA4 0710C 04F60406 (06212)
AA5 0710C 04F60406 (06213)
AA6 0710C 04F60406 (06214)
AA7 0710C 04F60406 (06215)
AA8 0710C 04F60406 (06216)
AA9 0710C 04F60406 (06217)
AAA 0710C 04F60406 (06218)
AAB 0710C 04F60406 (06219)
AAC 0710C 04F60406 (06220)
AAD 0710C 04F60406 (06221)
AAE 0710C 04F60406 (06222)
AAF 0710C 04F60406 (06223)
AAG 0710C 04F60406 (06224)
AAH 0710C 04F60406 (06225)
AAI 0710C 04F60406 (06226)
AAJ 0710C 04F60406 (06227)
AAK 0710C 04F60406 (06228)
AAL 0710C 04F60406 (06229)
AAO 0710C 04F60406 (06230)
AAP 0710C 04F60406 (06231)
AAQ 0710C 04F60406 (06232)
AAR 0710C 04F60406 (06233)
AAS 0710C 04F60406 (06234)
AAU 0710C 04F60406 (06235)
AAV 0710C 04F60406 (06236)
AAW 0710C 04F60406 (06237)
AAX 0710C 04F60406 (06238)
AAY 0710C 04F60406 (06239)
AAZ 0710C 04F60406 (06240)

```

```

MOV(DQ),ADD(A1,A5)\MOV(A4),ADD(A2,A5)
MOV(ZERO,A3)
MOV(A6),MOV(M1,M7)\MOV(A6),MUL(M1,M6)
MOV(A1),ADD(A3,A6)\MOV(A2),ADD(A4,A6)
MOV\MOV(R,DQ)
JUMP(DCOR$SA)

ENTRY FOR K+1 CLEAN-UP
CLEAR(M1)
MOV(TQA,M0)
MOV(A5),MUL(M0,M4)\MOV(A5),MUL(M0,M7)
MOV(A3),ADD(A1,A5)\MOV(A4),ADD(A2,A5)
MOV(A6),MUL(M0,M6)\MOV(A6),MUL(M0,M5)
MOV(DQ),ADD(A3,A6)\MOV(A2),ADD(A4,A6)
MOV(ZERO,A1)
MOV(TQA,M1)
MOV(A5),MUL(M1,M5)\MOV(A5),MUL(M1,M4)
JUMPS(DCEND,EO)
MOV(A3),ADD(A1,A5)\MOV(A4),ADD(A2,A5)
MOV(A6),MUL(M1,M7)\MOV(A6),MUL(M1,M6)
MOV(A1),ADD(A3,A6)\MOV(OQ),ADD(A4,A6)
MOV(ZERO,A2)
MOV(TQA,M0)
MOV(A5),MUL(M0,M6)\MOV(A5),MUL(M0,M5)
JUMPS(DCEND,EO)
MOV(DQ),ADD(A1,A5)\MOV(A4),ADD(A2,A5)
MOV(ZERO,A3)
MOV(A6),MUL(M0,M4)\MOV(A6),MUL(M0,M7)
MOV(A1),ADD(A3,A6)\MOV(A2),ADD(A4,A6)
MOV\MOV(R,DQ)
JUMP(DCOR$SA)

ENTRY FOR K+2 CLEAN-UP
CLEAR(M1)
MOV(TQA,M1)
MOV(A5),MUL(M1,M5)\MOV(A5),MUL(M1,M4)
MOV(A3),ADD(A1,A5)\MOV(A4),ADD(A2,A5)
MOV(A6),MUL(M1,M7)\MOV(A6),MUL(M1,M6)
MOV(DQ),ADD(A3,A6)\MOV(A2),ADD(A4,A6)
MOV(ZERO,A1)
MOV(TQA,M0)
MOV(A5),MUL(M0,M6)\MOV(A5),MUL(M0,M5)
JUMPS(DCEND,EO)
MOV(A3),ADD(A1,A5)\MOV(A4),ADD(A2,A5)
MOV(A6),MUL(M0,M4)\MOV(A6),MUL(M0,M7)
MOV(A1),ADD(A3,A6)\MOV(OQ),ADD(A4,A6)
MOV(ZERO,A2)
MOV(TQA,M1)
MOV(A5),MUL(M1,M7)\MOV(A5),MUL(M1,M6)

```

```

OUT=S3(M-1),S1(M+1),A4=S4(M-2),S2(M)
CLEAR S3 FOR NEW SUM
A6=PS3(M),PS1(M+2),A6=PS4(M-1),PS2(
A1=S1(M+1),S3(M),A2=S2(M),S54(M-1)
CONTINUE TO K+1 PHASE

MO=0(M)

OUT=S1(M-1)
CLEAR S1 FOR NEW SUM

M1=0(M+1)

JUMP OUT IF S1(M-1) IS LAST OUTPUT

OUT=S2(M-1)
CLEAR S2 FOR NEW SUM

MO=0(M+2)

JUMP OUT IF S2(M-1) IS LAST OUTPUT
OUT=S3(M-1)
CLEAR A3 FOR NEW S3 SUM

M1=0(M)

OUT=S1(M-1)
CLEAR A1 FOR NEW S1 SUM

MO=0(M+1)

JUMP OUT IF S1(M-1) FINAL OUTPUT

OUT=S2(M-1)
CLEAR A2 FOR NEW S2 SUM

M1=0(M+2)

```



```

PAGE 130:  (0000)(<C416>00016M,NS0.2, 29-Dec-00 14:30:40, Ed: WULF
APS3--ADOC APS PROGRAM FOR DISCRETE CORRELATION WITH AUTO ZERO FILL
(06220) * APS3-ADOC APS PROGRAM FOR DISCRETE CORRELATION WITH AUTO ZERO FILL
(06221) )BUFFER ATTRIBUTES FOR INITIAL TESTING
00000000 (06222) VBASE=0
00000032 (06223) VBASE=50
00000064 (06224) VBASE=100
(06225) )
0000000C (06226) VZ=12
00000005 (06227) VZ=5
00000007 (06228) VZ=15
00000007 (06229) VZ=2
00000002 (06230) VZ=2
00000002 (06231) VZ=2
00000002 (06232) VZ=2
(06233) )
(06234) ) BINDING DONE BY SBOCORZ
(06235) )
(06236) )
(06237) )
(06238) )
(06239) )
(06240) )
(06241) )
(06242) )
(06243) )
(06244) )
0725A 00000000 (06245) )
0725C 0000726C (06246) )
0725E 0000 (06247) )
0725F 000C (06248) )
07260 00000000 (06249) )
(06250) )
(06251) )
(06252) )
(06253) )
000 07262 00204240 (06254) )
001 07264 02300030 (06255) )
(06256) )
002 07266 04640032 (06257) )
003 07268 06700004 (06258) )
004 0726A 00220002 (06259) )
(06260) )
005 0726C 0A500002 (06261) )
(06262) )
006 0726E 0C390039 (06263) )
007 07270 0E190CF1 (06264) )
(06265) )
008 07272 10C2078A (06266) )
009 07274 17AA0002 (06267) )
00A 07276 14190CF1 (06268) )
00B 07278 16390001 (06269) )
(06270) )
00C 0727A 10440000 (06271) )
00D 0727C 10440000 (06272) )
(06273) )
(06274) )
(06275) )
(06276) )
(06277) )
(06278) )
(06279) )
(06280) )
(06281) )
(06282) )
(06283) )
(06284) )
(06285) )
(06286) )
(06287) )
(06288) )
(06289) )
(06290) )
(06291) )
(06292) )
(06293) )
(06294) )
(06295) )
(06296) )
(06297) )
(06298) )
(06299) )
(06300) )
(06301) )
(06302) )
(06303) )
(06304) )
(06305) )
(06306) )
(06307) )
(06308) )
(06309) )
(06310) )
(06311) )
(06312) )
(06313) )
(06314) )
(06315) )
(06316) )
(06317) )
(06318) )
(06319) )
(06320) )
(06321) )
(06322) )
(06323) )
(06324) )
(06325) )
(06326) )
(06327) )
(06328) )
(06329) )
(06330) )
(06331) )
(06332) )
(06333) )
(06334) )
(06335) )
(06336) )
(06337) )
(06338) )
(06339) )
(06340) )
(06341) )
(06342) )
(06343) )
(06344) )
(06345) )
(06346) )
(06347) )
(06348) )
(06349) )
(06350) )
(06351) )
(06352) )
(06353) )
(06354) )
(06355) )
(06356) )
(06357) )
(06358) )
(06359) )
(06360) )
(06361) )
(06362) )
(06363) )
(06364) )
(06365) )
(06366) )
(06367) )
(06368) )
(06369) )
(06370) )
(06371) )
(06372) )
(06373) )
(06374) )
(06375) )
(06376) )
(06377) )
(06378) )
(06379) )
(06380) )
(06381) )
(06382) )
(06383) )
(06384) )
(06385) )
(06386) )
(06387) )
(06388) )
(06389) )
(06390) )
(06391) )
(06392) )
(06393) )
(06394) )
(06395) )
(06396) )
(06397) )
(06398) )
(06399) )
(06400) )
(06401) )
(06402) )
(06403) )
(06404) )
(06405) )
(06406) )
(06407) )
(06408) )
(06409) )
(06410) )
(06411) )
(06412) )
(06413) )
(06414) )
(06415) )
(06416) )
(06417) )
(06418) )
(06419) )
(06420) )
(06421) )
(06422) )
(06423) )
(06424) )
(06425) )
(06426) )
(06427) )
(06428) )
(06429) )
(06430) )
(06431) )
(06432) )
(06433) )
(06434) )
(06435) )
(06436) )
(06437) )
(06438) )
(06439) )
(06440) )
(06441) )
(06442) )
(06443) )
(06444) )
(06445) )
(06446) )
(06447) )
(06448) )
(06449) )
(06450) )
(06451) )
(06452) )
(06453) )
(06454) )
(06455) )
(06456) )
(06457) )
(06458) )
(06459) )
(06460) )
(06461) )
(06462) )
(06463) )
(06464) )
(06465) )
(06466) )
(06467) )
(06468) )
(06469) )
(06470) )
(06471) )
(06472) )
(06473) )
(06474) )
(06475) )
(06476) )
(06477) )
(06478) )
(06479) )
(06480) )
(06481) )
(06482) )
(06483) )
(06484) )
(06485) )
(06486) )
(06487) )
(06488) )
(06489) )
(06490) )
(06491) )
(06492) )
(06493) )
(06494) )
(06495) )
(06496) )
(06497) )
(06498) )
(06499) )
(06500) )
(06501) )
(06502) )
(06503) )
(06504) )
(06505) )
(06506) )
(06507) )
(06508) )
(06509) )
(06510) )
(06511) )
(06512) )
(06513) )
(06514) )
(06515) )
(06516) )
(06517) )
(06518) )
(06519) )
(06520) )
(06521) )
(06522) )
(06523) )
(06524) )
(06525) )
(06526) )
(06527) )
(06528) )
(06529) )
(06530) )
(06531) )
(06532) )
(06533) )
(06534) )
(06535) )
(06536) )
(06537) )
(06538) )
(06539) )
(06540) )
(06541) )
(06542) )
(06543) )
(06544) )
(06545) )
(06546) )
(06547) )
(06548) )
(06549) )
(06550) )
(06551) )
(06552) )
(06553) )
(06554) )
(06555) )
(06556) )
(06557) )
(06558) )
(06559) )
(06560) )
(06561) )
(06562) )
(06563) )
(06564) )
(06565) )
(06566) )
(06567) )
(06568) )
(06569) )
(06570) )
(06571) )
(06572) )
(06573) )
(06574) )
(06575) )
(06576) )
(06577) )
(06578) )
(06579) )
(06580) )
(06581) )
(06582) )
(06583) )
(06584) )
(06585) )
(06586) )
(06587) )
(06588) )
(06589) )
(06590) )
(06591) )
(06592) )
(06593) )
(06594) )
(06595) )
(06596) )
(06597) )
(06598) )
(06599) )
(06600) )
(06601) )
(06602) )
(06603) )
(06604) )
(06605) )
(06606) )
(06607) )
(06608) )
(06609) )
(06610) )
(06611) )
(06612) )
(06613) )
(06614) )
(0
```


PAGE 140: (BRND)<OCAL6>BMMIGH.MSD.2, 29-Dec-88 14:38:48, R4: WOLF
AP53-ADOCOR APS PROGRAM FOR DISCRETE CORRELATION WITH AUTO ZERO FILL

```

A37 07200 66C207BA (06326)      LOAD(BR0,ZERO(1),TF)
      (06327) ,
A38 07202 70640032 (06328)      ADC$13: LOAD(BR2,YBASE(2))
A39 07204 72700004 (06329)      ADC$13A: LOAD(BR3,VZ-1)
A3A 07206 74220002 (06330)      ADC$13B: SUB(BR2,V1)
      (06331) ,
A3B 07208 76002260 (06332)      JUMP(ADC$10)
A3C 0720A 780A0006 (06333)      ADC$14: ADD(BR0,3*01)
A3D 0720C 7A010011 (06334)      MOV8(BR0,BR0)
A3E 0720E 7C090010 (06335)      MOV8(BR0,BR0)
A3F 07210 7E110033 (06336)      SUBL(BV1,3)
A40 07212 80190012 (06337)      MOV8(BR1,BV1)
A41 07214 82003060 (06338)      JUMP(D211A0)
      (06339) ,
A42 07216 84300032 (06340)      JOUTPUT PROGRAM
      (06341) ADC$15: SET(RA)
A43 07218 86400064 (06342)      ADC$16: LOAD(BW2,YBASE(2))
A44 0721A 8870000E (06343)      ADC$16A: LOAD(BW3,VZ-1)
A45 0721C 8A220002 (06344)      ADC$16B: SUB(BW2,V1)
      (06345) ,
A46 0721E 8CAB0002 (06346)      ADC$17: ADD(BW2,V1,TF)
A47 0721F 8E314601 (06347)      SUBL(BW3,1),JUMPP(ADC$17)
A48 07221 90000020 (06348)      NOP(0)
A49 07223 92000020 (06349)      NOP(0)
A4A 07225 94200030 (06350)      CLEAR(R0)
A4B 07227 96000020 (06351)      NOP(0)
A4C 07229 98000020 (06352)      NOP(0)
A4D 0722B 9A000020 (06353)      NOP(0)
      00000000 (06354)      ADC$A=0C
      00000000 (06355)      END
      0722E 0000009C (06356)      ADC$2=BL-ADC$
      (06357) ,
      (06358) ,
      (06359) ,
      (06360) ,

```

JASSIGN VALUE TO CHAIN

AD-A096 092

BOLT BERANEK AND NEWMAN INC CAMBRIDGE MA
DESIGN AND REAL-TIME IMPLEMENTATION OF A
DEC 80 J J WOLF, K D FIELD, W H RUSSELL

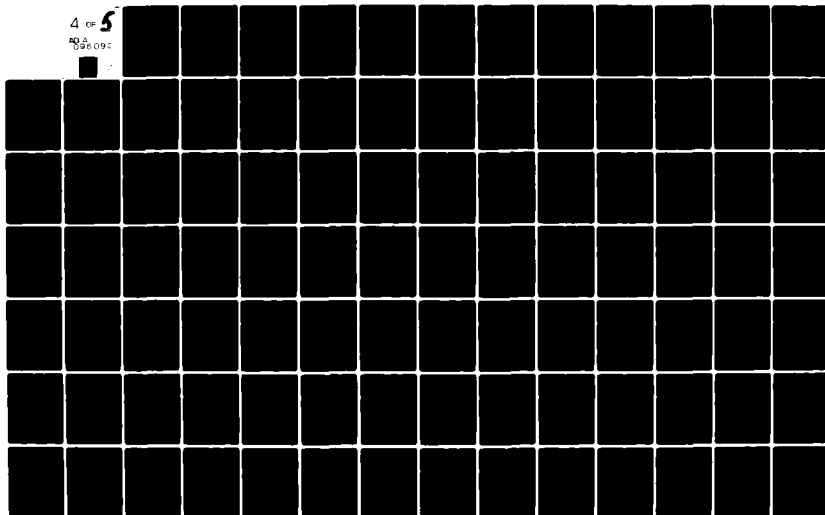
F/G 17/2.1
APC CODER FOR S--ETC(U)
DCA100-79-C-0037

UNCLASSIFIED

BBN-4565-VOL-2

NL

4 OF 5
NO
096092




```

(06361) - SBDORZ - SPECIAL BINDING MODULE FOR DCORZ
(06362) )
(06363) ) CALLING SEQUENCE
(06364) ) CALL R1,SBDORZ
(06365) )
(06366) ) WHERE (R1)=POINTER TO PCB NUMBER IN PCB
(06367) ) (R2)=POINTER TO ARRAY FUNCTION DISPATCH INFO
(06368) ) (R3)=ORIGIN FOR PENDING SUPPORT LIST
(06369) )
(06370) ) MODIFIED BY KFIELD 7/88
(06371) ) USE SAP'S FOR SAMPLE INCREMENTS
(06372) ) (TO HANDLE NEGATIVE SI'S).
(06373) )
(06374) SBDORZPUSHMIL R3,AFDT$ORG(R2)
(06375) MOVIR R7,-W$
(06376) PUSHMIL R3,0-W$(R3)
(06377) PUSHMIL R3,AFDT$ORG+W$(R2)
(06378) PUSHMIL R3,AFDT$ORG+2*W$(R2)
(06379) DECR R7,1
(06380) EVEN
(06381) PUSHMIL R3,0-2*W$(R3)
(06382) MOVIR R4,-3*W$(R3)
(06383) MOVIR R4,R4,MSK$SLBT
(06384) LRS R4,8
(06385) PUSHMIL R3,R4
(06386) MOVIR R5,R1
(06387) INCR R5,H$
(06388) MOVIR R4,R5,MSK$RBYT
(06389) PUSHMIL R3,R4
(06390) PUSHMIL R3,CLR$GGL
(06391) INCR R3,2
(06392) PUSHMIL R3,ZERO
(06393) MOVIR R3,AP$BSL
(06394) )
(06395) ) SCALAR BINDING
(06396) )
(06397) )
(06398) ) WZ BINDING
(06399) ) WZ=# OF LEADING ZEROS
(06400) )
(06401) MOVIR R4,H$(R1)
(06402) ANDIR R4,MSK$RBYT
(06403) ADDIR R4,ISVT$
(06404) MOVIR R3,R4
(06405) MOVIR R3,ADCS+W$*ADC$S<H$
(06406) MOVIR R3,ADCS+W$*ADC$2C<H$
(06407) )
(06408) ) USUPPER BINDING
(06409) ) USASE BINDING
(06410) MOVIR R6,2*W$(R1)
(06411) MOVIR R6,R6,MSK$SLBT
(06412) LRS R6,7
(06413) EVEN

(072FE C63408E8
(07300 907107FE
(07302 C68707FE
(07304 C63408EA
(07306 C63408EC
(07308 2771
(07309 8080
(0730A C48707FC
(0730C 904707FD
(0730E 70407F00
(07310 3C48
(07311 3438
(07312 4852
(07313 2651
(07314 704A00FF
(07316 3638
(07317 C6300792
(07319 2632
(0731A C630078A
(0731C F030074D

(0731E F0420801
(07320 9A4008FF
(07322 9C400502
(07324 6838
(07325 8030726D
(07327 E0307283

(07329 F0620802
(0732B 506C0700
(0732D 3C67

(06361) )
(06362) )
(06363) )
(06364) )
(06365) )
(06366) )
(06367) )
(06368) )
(06369) )
(06370) )
(06371) )
(06372) )
(06373) )
(06374) )
(06375) )
(06376) )
(06377) )
(06378) )
(06379) )
(06380) )
(06381) )
(06382) )
(06383) )
(06384) )
(06385) )
(06386) )
(06387) )
(06388) )
(06389) )
(06390) )
(06391) )
(06392) )
(06393) )
(06394) )
(06395) )
(06396) )
(06397) )
(06398) )
(06399) )
(06400) )
(06401) )
(06402) )
(06403) )
(06404) )
(06405) )
(06406) )
(06407) )
(06408) )
(06409) )
(06410) )
(06411) )
(06412) )
(06413) )

)STACK APV MODULE...
)STACK APV MODULE S.A. AND SIZE
)STACK APS MODULE BUS ORIGIN
)STACK CSPU MODULE BUS ORIGIN

)APS MODULE SIZE
)GET SPECIAL SUPPORT SEMAPHORE

)STACK SPECIAL SUPPORT SEMAPHORE
)POINT TO FIRST SCALAR ID

)EXTRACT SA ID
)STACK SA ID
)RESET 6 FLAGS

)SET FOR NO PDP
)POINT TO START OF SUPPORT LIST

)GET SCALAR WZ ID
)MASK RIGHT BYTE
)R4=R4+ISVT$
)R3=WZ
)STORE IN APS MAIN MEMORY

)GET U ID
)GET PROPER BYTE
)CONVERT TO BCT$ INDEX

```

```

PAGE 142: (08ND)C0CA1G08M16M.N50.2, 29-Dec-88 14:30:40, Ed: WOLF
SDC0RZ - SPECIAL BINDING MODULE FOR D0RZ

0732E C04C0502 (06414)      MOVNRL R4,BCT$BA(R6)
07330 F070727A (06415)      MOVNR R7,ADC$W$*ADC$2
07332 564EFFF0 (06416)      TORNR R4,R7,$FFF0
07334 0440727A (06417)      MOVNRL R4,ADC$W$*ADC$2
07340 (06418)
07341 (06419)
07342 (06420)
07343 (06421)
07344 (06422)
07345 (06423)
07346 (06424)
07347 (06425)
07348 (06426)
07349 (06427)
07350 (06428)
07351 (06429)
07352 (06430)
07353 (06431)
07354 (06432)
07355 (06433)
07356 (06434)
07357 (06435)
07358 (06436)
07359 (06437)
07360 (06438)
07361 (06439)
07362 (06440)
07363 (06441)
07364 (06442)
07365 (06443)
07366 (06444)
07367 (06445)
07368 (06446)
07369 (06447)
07370 (06448)
07371 (06449)
07372 (06450)
07373 (06451)
07374 (06452)
07375 (06453)
07376 (06454)
07377 (06455)
07378 (06456)
07379 (06457)
07380 (06458)
07381 (06459)
07382 (06460)
07383 (06461)
07384 (06462)
07385 (06463)
07386 (06464)
07387 (06465)
07388 (06466)

; (U2-1) BINDING
; (U2-1) = USIZE - 1 = U $S
MOVNR R4,HS*BCT$AD(R6)
MOVNR R4,ADC$W$*ADC$2A*HS

; UI BINDING
; UI=0 SAMPLE INCREMENT
MOVNR R5,BCT$AD(R6)
MOVNR R7,RS
SAP R5,ADC$W$*ADC$2B
SAP R5,ADC$W$*ADC$5
SAP R5,ADC$W$*ADC$7
SAP R5,ADC$W$*ADC$8
SAP R5,ADC$W$*ADC$9

; 4*UI BINDING
; 4*UI=4*(0 SAMPLE INCREMENT)
LLS R5,2
SAP R5,ADC$W$*ADC$11

; M2*UI BINDING
; (M2*UI) = (0 OF LEADING ZEROS) * (U $I)
MOVNR R3,R7
LLS R3,1
SAP R3,ADC$W$*ADC$3
SAP R3,ADC$W$*ADC$4

; V BUFFER BINDING
; VBASE BINDING
MOVNR R6,3*HS(R1)
MOVNR R6,R6,MSK$LBVT
LLS R6,7
EVEN
MOVNRL R4,BCT$BA(R6)
MOVNR R7,ADC$W$*ADC$1
TORNR R4,R7,$FFF0
MOVNRL R4,ADC$W$*ADC$1

MOVNRL R3,BCT$BA(R6)
MOVNR R5,ADC$W$*ADC$10
TORNR R3,R5,$FFF0
MOVNRL R3,ADC$W$*ADC$10

MOVNRL R4,BCT$BA(R6)
MOVNR R7,ADC$W$*ADC$13
TORNR R4,R7,$FFF0
MOVNRL R4,ADC$W$*ADC$13

; RS=U $I
; R7=RS=U $I
; MOVE TO APS MAIN MEMORY
; MOVE TO APS MAIN MEMORY
; MOVE TO APS MAIN MEMORY
; MOVE TO APS MAIN MEMORY
; MOVE TO APS MAIN MEMORY
; MOVE TO APS MAIN MEMORY
; RS=SI .L. 2=4*UI
; MOVE TO APS MAIN MEMORY

; R3=2*M2*UI
; R3=M2*UI
; MOVE TO APS MAIN MEMORY
; MOVE TO APS MAIN MEMORY

; GET V ID
; GET PROPER BYTE
; CONVERT TO BCT$ INDEX
; GET 17-BIT VBASE ADDR
; APS ADDR IN MAIN MEMORY
; MOVE TO APS IN MAIN MEMORY

; GET 17-BIT VBASE ADDR
; APS ADDR IN MAIN MEMORY
; MOVE TO APS IN MAIN MEMORY

; GET 17-BIT VBASE ADDR
; APS ADDR IN MAIN MEMORY
; MOVE TO APS IN MAIN MEMORY

```

PAGE 143: [0800]OCAL16>08016M.NSO.2, 29-Dec-88 14:38:40, Ed: WOLP
SBOCORZ - SPECIAL BINDING MODULE FOR DCORZ

```

(06467) ; (VZ-1) BINDING
(06468) ; (VZ-1)=VSIZE-1-V NS
0736E F03C0605 MOVNR R3,NS+BCTSAD(R6)
07370 E0307269 MOVNR R3,ADC$+W$+ADC$1A+H$
07372 E03072A9 MOVNR R3,ADC$+W$+ADC$1BA+H$
07374 E03072D5 MOVNR R3,ADC$+W$+ADC$13A+H$
(06473) ;
(06474) ; VI BINDING
(06475) ; VI=V SAMPLE INCREMENT
07376 F04C0604 MOVNR R4,BCTSAD(R6)
07378 EF40726A SAF R4,ADC$+W$+ADC$1B
0737A EF407274 SAF R4,ADC$+W$+ADC$1B2
0737C EF407292 SAF R4,ADC$+W$+ADC$6
0737E EF4072AA SAF R4,ADC$+W$+ADC$10B
07380 EF4072C4 SAF R4,ADC$+W$+ADC$12
07382 EF4072D6 SAF R4,ADC$+W$+ADC$13B
(06483) ; YBUFFER BINDING
(06484) ; YBASE BINDING
(06485) ;
07384 F0620001 MOVNR R6,MS(R1)
07386 506CFF00 MOVNR R6,R6,MSR$LOVT
07388 3C67 LRS R6,7
07389 8000 EVEN
0738A C04C0502 MOVNRL R4,BCTSBA(R6)
0738C F07072E0 MOVNR R7,ADC$+W$+ADC$16
0738E 564PFF0 MOVNR R4,R7,$PFF0
07390 044072E0 MOVNRL R4,ADC$+W$+ADC$16
(06494) ;
(06495) ; (VZ-1) BINDING
(06496) ; VZ-1 = VSIZE - 1 = BS
07392 F07C0605 MOVNR R7,HS+BCTSAD(R6)
07394 E07072E0 MOVNR R7,ADC$+W$+ADC$16A+H$
(06499) ;
(06500) ; VI BINDING
(06501) ; VI = SAMPLE INCREMENT
07396 F03C0604 MOVNR R3,BCTSAD(R6)
07398 EF3072EC SAF R3,ADC$+W$+ADC$16B
0739A EF3072EE SAF R3,ADC$+W$+ADC$17
(06505) ;
(06506) ;
0739C 0000FF63 JMP AP$BNDRO

JGET BS = # COLS - 1
JMOVE TO APS MAIN MEMORY
JMOVE TO APS MAIN MEMORY
JMOVE TO APS MAIN MEMORY

JH4 = V SI
JMOVE TO APS MAIN MEMORY
JMOVE TO APS MAIN MEMORY
JMOVE TO APS MAIN MEMORY
JMOVE TO APS MAIN MEMORY
JMOVE TO APS MAIN MEMORY
JMOVE TO APS MAIN MEMORY

JGET Y ID
JGET PROPER BYTE
JCONVERT TO BCT$ INDEX
JGET 17-BIT YBASE ADDR
JAPS ADR IN MAIN MEM
JMODIFIED APS INSTR IN MAIN MEM

JGET BS = # OF COLS - 1
JMOVE TO APS MAIN MEMORY

JR3 = Y SI
JMOVE TO APS MAIN MEMORY
JMOVE TO APS MAIN MEMORY

```

```

(06500) * MPIFF$ MODULE TO PROCESS THE MPIFF(ISA,ISB,FLID) FCB
(06509) * MPFIELD 9/79
(06510) *
(06511) *
(06512) * EXECUTES FUNCTION LIST 'FLID' IF (ISA .NE. 0) .AND.
(06513) * (ISB .EQ. 0)
(06514) *
(06515) * FCB FORMAT (16 BIT FORMAT SHOWN)
(06516) *
(06517) *
(06518) *
(06519) *
(06520) *
(06521) *
(06522) *
(06523) *
(06524) *
(06525) *
(06526) *
(06527) *
(06528) *
(06529) *
(06530) *
(06531) *
(06532) *
(06533) *
(06534) * MPIFF$
(06535) *
(06536) *
(06537) *
(06538) *
(06539) *
(06540) *
(06541) *
(06542) *
(06543) *
(06544) *
(06545) *
(06546) *
(06547) *
(06548) *
(06549) *
(06550) *

WORD LEFT BYTE RIGHT BYTE
-----
0: PTR TO NEXT FCB AND FUNCTION LIST FLAG (LSB)
1: 105 ISA
2: 0 ISB
3: 0 FLID
4: 0
5: 0

PROGRAM REGISTER USAGE:
R1: PTR TO WORD 1 OF FCB ENTRY (SET ON ENTRY)
R2: FUNCTION LIST ID
R4: SCALAR A ID
R5: SCALAR B ID

EVEN MOVK R4,R1,MSK$RBYT
MOVNR R5,MS(R1)
MOVNR R2,MS(R1)

CMPNZ ISVTS(R4)
JMP IFF$1,NEZ
RETURN
EVEN

CMPNZ ISVTS(R5)
JMP XFL$01,EQZ

RETURN

EVEN

0739E 704200FF (06534) MPIFF$
073A0 70520001 (06535)
073A2 70220002 (06536)
073A4 22000502 (06537)
073A6 011073AA (06538)
073A8 0E70 (06539)
073A9 0800 (06540)
073AA 22000502 (06541)
073AC 0010192C (06542)
073AE 0E70 (06543)
073AF 0800 (06544)
073B0 0000 (06545)
073B2 0000 (06546)
073B4 0000 (06547)
073B6 0000 (06548)
073B8 0000 (06549)
073BA 0000 (06550)

```

PAGE 145: (0000)DCAL6>BBNICH.MSD.2, 29-Dec-88 14:38:48, 24: VOL7
 MPCSC -- G-FLAG SET/CLEAR

```

(06551) * MPCSC -- G-FLAG SET/CLEAR
(06552) JJW, 18 OCT 79
(06553) *
(06554) * G-FLAG NUMBER IN 1(01)
(06555) * 0/1 = CLEAR/SET IN 2(01)
(06556) *
(06557) MPCSC: MOVNR R2,1(01)
(06558) INCX R2,4
(06559) MOVNR R3,2(01)
(06560) LLS R3,5
(06561) TORNR R2,R3,R20
(06562) MOVNR R2,SYSPFGS
(06563) RETURN
(06564) EVEN

07300 70220001
07302 2624
07303 70320002
07305 3A35
07306 56360020
07308 E021FFCE
0730A 0270
0730B 0000

JGET G-FLAG NUMBER
JADD OFFSET FOR SYSPFGS
JGET CLR/SET BIT
JSHIFT TO PROPER POSITION
JAND OFF AND PUT IN R2
JDD THE MOVE THAT SETS/CLRS IT
  
```

```

(00565) "DEBUGGING "BREAKPOINT" FACILITY
(00566) )
(00567) ) TO HALT CPU EXECUTION AND SAVE REGISTERS ON THE STACK,
(00568) ) PATCH: "CALL RI, BKPT" (AS ASSEMBLED BELOW)
(00569) ) INTO THE DESIRED BREAKPOINT LOCATION (USING WPOK).
(00570) )
(00571) ) CALL RI, BKPT
(00572) BKPT: JMP BKPT
0730C 0010730E
0730E 0000730E

```

ROIMINIAO 'SAIINCXKZ JO JOJ. (CE590)

(06573) -T
(06574) -
(06575) -
(06576) -
(06577) -
(06578) -
(06579) -

0000730 (06576)
TUE\$CUB = IL

0360 (0659) 093

ADCS:	07262	(00221)	(06245)	(06253)	(06356)	(06405)	(06406)	(06415)	(06417)	(06422)	(06420)
		(06429)	(06430)	(06431)	(06432)	(06437)	(06443)	(06444)	(06453)	(06455)	(06450)
		(06460)	(06463)	(06465)	(06470)	(06471)	(06472)	(06477)	(06470)	(06479)	(06400)
		(06481)	(06482)	(06491)	(06493)	(06490)	(06503)	(06504)			
ADCS1:	00002	(06257)	(06453)	(06455)							
ADCS10:	00022	(06300)	(06316)	(06332)	(06450)	(06460)					
ADCS10A:	00023	(06301)	(06471)								
ADCS10B:	00024	(06302)	(06400)								
ADCS11:	00026	(06305)	(06437)								
ADCS12:	00031	(06320)	(06401)								
ADCS13:	00030	(06320)	(06463)	(06465)							
ADCS13A:	00039	(06329)	(06472)								
ADCS13B:	0003A	(06330)	(06402)								
ADCS14:	0003C	(06333)									
ADCS15:	00042	(06254)	(06341)								
ADCS16:	00043	(06342)	(06491)	(06493)							
ADCS16A:	00044	(06343)	(06490)								
ADCS16B:	00045	(06344)	(06503)								
ADCS17:	00046	(06346)	(06347)	(06504)							
ADCS1A:	00003	(06250)	(06470)								
ADCS1B:	00004	(06259)	(06477)								
ADCS1B2:	00009	(06267)	(06470)								
ADCS2:	0000C	(06264)	(06200)	(06272)	(06415)	(06417)					
ADCS2A:	0000D	(06273)	(06472)								
ADCS2B:	0000E	(06274)	(06420)								
ADCS2C:	00010	(06276)	(06406)								
ADCS3:	00012	(06200)	(06443)								
ADCS4:	00014	(06202)	(06444)								
ADCS5:	00017	(06207)	(06479)								
ADCS6:	00010	(06200)	(06479)								
ADCS7:	00010	(06294)	(06430)								
ADCS8:	0001F	(06296)	(06431)								
ADCS9:	00021	(06290)	(06432)								
ADCSA:	00000	(06240)	(06354)								
ADCSB:	00005	(06245)	(06261)	(06405)							
ADCSZ:	0009C	(06247)	(06356)	(02030)							
ADPA:	065FE	(00161)	(02007)								
ADPA\$A:	0660A	(02001)	(02027)								
ADPA\$B:	0661A	(02797)	(02029)								
ADPA\$Z:	00020	(02000)	(02030)								
ADPAI:	065FE	(02000)									
ADPAO:	0000A	(02009)	(02022)								
ADPE:	065D2	(00160)	(02761)								
ADPE\$A:	00000	(02750)	(02762)	(02700)							
ADPE\$Z:	00012	(02759)	(02700)								
ADP\$ORC:	00000	(00169)	(00124)	(00129)	(00134)	(00139)	(00144)	(00149)	(00154)	(00159)	(00164)
		(00169)	(00174)	(00179)	(00184)	(00189)	(00194)	(00199)	(00204)	(00209)	(00214)
		(00219)	(00203)	(02106)	(02107)	(04704)	(04707)	(04708)	(05369)	(05372)	(05373)
		(06374)	(06377)	(06378)							
AP\$A\$S:	00245	(00023)									
AP\$ANDR:	0007A	(00024)	(03595)								
AP\$ANDR0:	00F63	(00025)	(02203)	(05075)	(05509)	(06507)					
AP\$ANDR1:	00F20	(00026)									

PAGE 149:

AP 5051:

APCSW41:	00041 (04057) (04957) (04960) (04962)
APCSW51:	00032 (04053) (04966) (04968) (04971)
APCSW7:	00042 (04787) (04821) (04824)
APCSZ1:	00008 (04493) (04765)
APCU1:	00072 (00190) (04032)
APCU811:	00091 (04350) (04363)
APCU812:	00009 (04315) (04346)
APCU831:	00004 (04323) (04337)
APCUS0UM:	00095 (04333) (04342) (04359) (04373)
APCUS5A1:	00000 (04029) (04033) (04419)
APCUS5Z1:	000A0 (04030) (04419)
APCU\$TOP:	00015 (04083) (04399)
APSSMD1:	00004 (00030)
APSLA:	1FFC0 (00039) (01158) (01670)
APSLDC:	1FFC0 (00040)
APSMEN:	1FFC0 (00041) (02712) (04697) (05055)
APSR:	1FFC0 (00042)
BCT\$AD:	00604 (00044) (02229) (05404) (05421) (05426) (05465) (05482) (05499)
	(06421) (06426) (06476) (06502)
BCT\$AT:	00606 (00045) (03500) (02241) (02251) (02256) (02266) (02271) (04020) (04032) (04044)
BCT\$BN:	00502 (00046) (02223) (04062) (04071) (04080) (04089) (04090) (04097) (04098) (04099) (04901) (04906) (05006) (05025) (05034) (05043) (05052) (05061) (05070) (05390) (05415) (05437) (05454) (05476) (05493)
	(06414) (06452) (06457) (06462) (06490)
BITSGR:	00000 (00047)
BITSRC:	00000 (00048)
BIT\$N2:	00003 (00049) (04023)
BEPT:	07302 (06571) (06572)
CF40B\$:	01AD6 (00051)
CLR\$G61:	00792 (00052) (02199) (04800) (05305) (06390)
COS\$:	03190 (00053) (03591) (03700)
COS\$B:	00024 (02390) (02401) (02404) (02412)
CSPU\$R03:	021FC (00054) (00127) (00132) (00137) (00152) (00157) (00162) (00167) (00172) (00177)
	(00197) (00207) (00212) (00217)
CTMCL:	09000 (00239) (00240) (00241) (02470)
CTMC2:	00010 (00240) (00242) (02402)
CTMC3:	09000 (00241) (02406)
CTHDG:	099F0 (00249) (00250) (03492) (03498) (03504)
CTHGI:	09970 (00240) (00249) (03486)
CTHF1:	09830 (00242) (00243) (01596)
CTHF2:	09800 (00243) (00244)
CTHF3:	098F0 (00244) (00245)
CTHF4:	09910 (00245) (00246)
CTHF5:	09930 (00246) (00247)
CTHF6:	09950 (00247) (00248)
CTHU:	09970 (00250) (00251) (04629) (04643) (04655)
D211A5:	00020 (06393) (06312)
D211A6:	0002C (06395) (06313)
D211A7:	00020 (06297) (06314)
D211A8:	00030 (06206) (06319) (06321) (06330)
DCEMD:	00007 (06061) (06114) (06120) (06139) (06147) (06166) (06174) (06193) (06201) (06211) (06212)
DCLP:	00026 (06073) (06102)

BCL1:	00030 (06001) (06129)	
BCL2:	00054 (06009) (06156)	
BCL3:	00060 (06097) (06183)	
OCOR5:	07150 (00220) (06051)	
DCOR52A:	00000 (06040) (06053)	(06126) (06153) (06180) (06287) (06213) (06214)
DCOR52:	00005 (06049) (06214)	
OCOR0:	00009 (06067) (06099)	
OCOR1:	00010 (06075)	
OCOR2:	00017 (06083)	
OCOR3:	0001E (06091)	
DEAL3:	070FA (00216)	(05937) (05942) (06014)
DEAL5A:	07132 (05940)	(06000)
DEAL5B:	0713A (05936)	(06012)
DEAL5D:	00000 (05943) (05967)	
DEAL5S:	0000C (05937) (05969)	
DEAL5Z:	00056 (05939) (06014)	
DEAL0:	07020 (00215) (05913)	
DEAL0SA:	00000 (05910) (05914)	(05916) (05920)
DEAL0SZ:	00004 (05911) (05920)	
DW5:	00794 (00056) (01530)	(02714) (02715) (05669) (05670) (05057) (05058)
DUMP3:	06660 (00166) (02090)	(02919)
DUMPSA:	06672 (02093) (02917)	
DUMPS1:	0667A (02089) (02910)	
DUMPS2:	0001E (02092) (02919)	
DUMPS3:	00000 (02099)	
DUMFS0:	00004 (02900) (02907)	
DUMPU:	06620 (00165) (02043)	
DUMPU5A:	00000 (02040) (02044)	(02000)
DUMPU5Z:	0001C (02041) (02000)	
DVLC1:	09100 (00251) (00252)	(00253) (02479)
DVLC2:	09110 (00252) (00254)	(02403)
DVLC3:	09100 (00253) (02407)	
DVLC6:	090F0 (00261) (00262)	(03493) (03499) (03505)
DVLC:	09130 (00260) (00261)	(03407)
DVLR1:	09130 (00254) (00255)	(01597)
DVLR2:	09100 (00255) (00256)	
DVLF3:	091F0 (00256) (00257)	
DVLF4:	09110 (00257) (00250)	
DVLF5:	09030 (00250) (00259)	
DVLF6:	09050 (00259) (00260)	
DVLF:	090F0 (00262) (00263)	(04665)
DVLF:	09C00 (00263)	
LABOR:	011FA (00050)	
FMID:	0000A (02760)	(02772)
F1:	00000 (03794)	(03797)
F1A3:	03110 (00060)	
F1AL5:	069F0 (00101)	(00106) (03700) (03069)
F1AL3:	0001C (03706)	(03023)
F1AL5A:	0614C (03777)	(03050)
F1AL5C:	06100 (03773)	(03066)
F1AL5Z:	00000 (03776) (03069)	
F2:	00012 (03002) (03007)	
F4:	0002D (03034) (03045)	(03051)

FDY\$:	007E0 (00061) (00220) (00231)
FDP\$1:	06900 (03574) (03584)
FOP\$SM:	06906 (00102) (03579)
FOP\$SM:	06902 (00187) (03573)
FOS\$1:	03040 (00062) (00185) (03612)
FOSL\$1:	00012 (03625) (03642)
FOSL\$2:	00019 (03630) (03653) (03694)
FOSL\$3:	0003A (03649) (03701) (03742)
FOSL\$4:	00059 (03697) (03744)
FOSL\$5:	00000 (03609) (03617) (03754) (03756)
FOSL\$5:	00064 (03610) (03756) (03757)
FYS\$CS:	0079A (00063) (03579)
FLC\$CL:	00000 (00064) (03573) (03579)
FLC\$CL:	00004 (00065) (03573)
FLC\$CL:	00005 (00066)
FLC\$CL:	00006 (00067)
FLC\$CL:	00007 (00068)
FLC\$CL:	00011 (00069)
FLC\$SE:	00020 (00070) (03573)
FLC\$SE:	00008 (04487) (04520)
FLC\$SE:	06764 (00175) (03221)
FLC\$SE:	00042 (03267) (03283)
FLC\$SE:	00052 (03250) (03273) (03326)
FLC\$SE:	00055 (03352) (03358)
FLC\$SE:	00058 (03360) (03362)
FLC\$SE:	0005F (03357) (03364)
FLC\$SE:	00061 (03227) (03231)
FLC\$SE:	00065 (03370) (03396)
FLC\$SE:	00077 (03381) (03403)
FLC\$SE:	00079 (03386) (03406)
FLC\$SE:	00078 (03391) (03409)
FLC\$SE:	00080 (03210) (03223)
FLC\$SE:	00084 (03219) (03421)
FLC\$SE:	00074 (00176) (03445)
FLC\$SE:	00080 (03473) (03480)
FLC\$SE:	0008C (03475) (03476)
FLC\$SE:	0008C (03448) (03556)
FLC\$SE:	0008F (03444) (03560)
FLC\$SE:	00032 (03457) (03534)
FLC\$SE:	00020 (03459) (03514)
FLC\$SE:	00003 (03445) (03462)
FLC\$SE:	0008F (03447) (03562)
FLC\$SE:	0003C (00072)
FLC\$SE:	00042 (00196) (05140) (05217)
FLC\$SE:	0007C (05143) (05211)
FLC\$SE:	00084 (05139) (05215)
FLC\$SE:	00017 (05146) (05193)
FLC\$SE:	00002 (05140) (05151)
FLC\$SE:	00056 (05142) (05217)
FLC\$SE:	00032 (00195) (05116)
FLC\$SE:	00000 (05113) (05117) (05119) (05123)
FLC\$SE:	00004 (05114) (05123)

```

MS:
00001 (00074) (01153) (01644) (01669) (02191) (02196) (02206) (02210) (02213) (02216)
(02219) (02229) (02231) (02233) (02234) (02235) (02237) (02247) (02262) (02710)
(03500) (03500) (04693) (04700) (04731) (04792) (04800) (04812) (04816) (04816)
(04820) (04839) (04917) (04975) (04994) (05013) (05377) (05382) (05394) (05411)
(05421) (05422) (05433) (05438) (05450) (05460) (05461) (05462) (05472) (05489)
(05499) (05500) (05501) (05853) (06382) (06387) (06401) (06405) (06406) (06410)
(06421) (06422) (06440) (06469) (06470) (06471) (06472) (06486) (06497) (06498)
(06535)
06700 (00171) (03103) (03165)
00002 (03109)
00006 (03115)
00010 (03130) (03131)
00012 (03132)
00016 (03147)
0001A (03152) (03153)
0001C (03154)
06740 (03089) (03160)
06746 (03085) (03164)
00015 (03105) (03144)
0005A (03080) (03165)
06680 (00170) (02971)
00039 (03026) (03027)
00020 (02900) (03060)
00000 (02960) (02976)
00040 (02969) (03060)
00030 (02010)
073AA (06539) (06543)
00000 (00360)
06202 (01600) (01669) (01675)
0658A (02651) (02725)
065C2 (02647) (02726)
06584 (02667) (02680) (02710)
00003 (02640) (02664)
0006E (02650) (02727)
06562 (00156) (02640) (02660) (02727)
00024 (02661) (02700)
0651A (00155) (02592) (02620)
00000 (02589) (02593)
00020 (02590) (02620)
00502 (00076) (02215)
0630A (02017) (02152)
0005F (01417) (01423)
0004E (01451) (01509)
00070 (01509) (01635)
00066 (01591) (01619)
00053 (01382) (01400)
0004P (00434) (00440)
0107E (00070)
00010 (00395) (00445)
00000 (00084) (00507)
(00574) (00582) (00583) (00584) (00587) (00592) (00593) (00594) (00597) (00598)
(00600) (00601) (00602) (00603) (00604) (00605) (00606) (00607) (00608) (00609)
(00610) (00611) (00612) (00613) (00614) (00615) (00616) (00617) (00618) (00619)
(00620) (00621) (00622) (00623) (00624) (00625) (00626) (00627) (00628) (00629)
(00630) (00631) (00632) (00633) (00634) (00635) (00636) (00637) (00638) (00639)
(00640) (00641) (00642) (00643) (00644) (00645) (00646) (00647) (00648) (00649)
(00650) (00651) (00652) (00653) (00654) (00655) (00656) (00657) (00658) (00659)
(00660) (00661) (00662) (00663) (00664) (00665) (00666) (00667) (00668) (00669)
(00670) (00671) (00672) (00673) (00674) (00675) (00676) (00677) (00678) (00679)
(00680) (00681) (00682) (00683) (00684) (00685) (00686) (00687) (00688) (00689)
(00690) (00691) (00692) (00693) (00694) (00695) (00696) (00697) (00698) (00699)
(00700) (00701) (00702) (00703) (00704) (00705) (00706) (00707) (00708) (00709)
(00710) (00711) (00712) (00713) (00714) (00715) (00716) (00717) (00718) (00719)
(00720) (00721) (00722) (00723) (00724) (00725) (00726) (00727) (00728) (00729)
(00730) (00731) (00732) (00733) (00734) (00735) (00736) (00737) (00738) (00739)
(00740) (00741) (00742) (00743) (00744) (00745) (00746) (00747) (00748) (00749)
(00750) (00751) (00752) (00753) (00754) (00755) (00756) (00757) (00758) (00759)
(00760) (00761) (00762) (00763) (00764) (00765) (00766) (00767) (00768) (00769)
(00770) (00771) (00772) (00773) (00774) (00775) (00776) (00777) (00778) (00779)
(00780) (00781) (00782) (00783) (00784) (00785) (00786) (00787) (00788) (00789)
(00790) (00791) (00792) (00793) (00794) (00795) (00796) (00797) (00798) (00799)
(00800) (00801) (00802) (00803) (00804) (00805) (00806) (00807) (00808) (00809)
(00810) (00811) (00812) (00813) (00814) (00815) (00816) (00817) (00818) (00819)
(00820) (00821) (00822) (00823) (00824) (00825) (00826) (00827) (00828) (00829)
(00830) (00831) (00832) (00833) (00834) (00835) (00836) (00837) (00838) (00839)
(00840) (00841) (00842) (00843) (00844) (00845) (00846) (00847) (00848) (00849)
(00850) (00851) (00852) (00853) (00854) (00855) (00856) (00857) (00858) (00859)
(00860) (00861) (00862) (00863) (00864) (00865) (00866) (00867) (00868) (00869)
(00870) (00871) (00872) (00873) (00874) (00875) (00876) (00877) (00878) (00879)
(00880) (00881) (00882) (00883) (00884) (00885) (00886) (00887) (00888) (00889)
(00890) (00891) (00892) (00893) (00894) (00895) (00896) (00897) (00898) (00899)
(00900) (00901) (00902) (00903) (00904) (00905) (00906) (00907) (00908) (00909)
(00910) (00911) (00912) (00913) (00914) (00915) (00916) (00917) (00918) (00919)
(00920) (00921) (00922) (00923) (00924) (00925) (00926) (00927) (00928) (00929)
(00930) (00931) (00932) (00933) (00934) (00935) (00936) (00937) (00938) (00939)
(00940) (00941) (00942) (00943) (00944) (00945) (00946) (00947) (00948) (00949)
(00950) (00951) (00952) (00953) (00954) (00955) (00956) (00957) (00958) (00959)
(00960) (00961) (00962) (00963) (00964) (00965) (00966) (00967) (00968) (00969)
(00970) (00971) (00972) (00973) (00974) (00975) (00976) (00977) (00978) (00979)
(00980) (00981) (00982) (00983) (00984) (00985) (00986) (00987) (00988) (00989)
(00990) (00991) (00992) (00993) (00994) (00995) (00996) (00997) (00998) (00999)

```


NULP3A:	06100 (01441) (01650)	
NULP3AP:	06080 (00141) (01430)	(01440) (01650)
NULP3APB:	06016 (00140) (01230)	
NULP3B:	06162 (01437) (01656)	
NULP3C:	00002 (01430) (01450)	(01454)
NULP3SA:	00000 (01234) (01241)	(01430)
NULP3SZ:	00065 (01235) (01430)	
NULP3Z:	0010A (01440) (01650)	
NULP3Z:	00039 (01342) (01300)	
NULP3Z:	00051 (01326) (01307)	
NULP3Z:	00040 (01367) (01301)	
NULP3Z:	00017 (01277) (01306)	
NULP3Z:	0004P (01256) (01305)	(01305)
NULP3Z:	00010 (01250) (01267)	
NULP3Z:	0004R (01561) (01567)	
NULP3Z:	0004C (01549) (01569)	
NULP3Z:	00032 (01532) (01537)	
NULP3Z:	0002A (01499) (01527)	
NULP3Z:	001F2 (00142) (01661)	
NULP3Z:	00015 (01475) (01403)	(01405)
NULP3Z:	00012 (01476) (01400)	
NULP3Z:	00020 (01509)	
NULP3Z:	06170 (01662) (01660)	
NULP3Z:	06310 (02039) (02042)	
NULP3Z:	00002 (06229) (06261)	(06276) (06280) (06282)
NULP3Z:	00010 (00006) (00656)	(00671) (00934) (00943)
NULP3Z:	00010 (05754)	(02601) (02605) (04062) (05750)
PIT3S:	052A4 (00136) (01023)	(01040) (01179)
PIT3S:	05F22 (01101) (01116)	(01153)
PIT3S:	06007 (01026) (01174)	
PIT3S:	06000 (01022) (01170)	
PIT3S:	00026 (01043) (01143)	
PIT3S:	00001 (01023) (01046)	
PIT3S:	00070 (01025) (01179)	
PIT3S:	070C0 (05000) (05072)	
PIT3S:	070D6 (05796) (05073)	
PIT3S:	07098 (05015) (05020)	(05853)
PIT3S:	00003 (05797) (05812)	
PIT3S:	00070 (05799) (05074)	
PIT3S:	07076 (00211) (05797)	(05800) (05874)
PIT3S:	00020 (05009) (05051)	
PIT3S:	07022 (00210) (05741)	
PIT3S:	00000 (05730) (05742)	(05777)
PIT3S:	00020 (05739) (05777)	
PIT3S:	052F4 (00135) (00040)	
PIT3S:	00007 (00052) (00060)	
PIT3S:	00010 (00050) (00063)	
PIT3S:	00015 (00050) (00070)	(00082)
PIT3S:	00010 (00054) (00065)	(00075)
PIT3S:	00010 (00056) (00080)	
PIT3S:	0001P (00067) (00086)	
PIT3S:	00025 (00080) (00092)	
PIT3S:	00027 (00091) (00090)	

PRISON:

151 151

Topic

PAGE 159: [0000]UCAL6>00016M.WSN.2, 29-Dec-88 14:30:40, Ed: WOLF
TOP OF EXECUTIVE DEFINITION

YOUS:	06303 (01999)	(06349) (06346)	(03047) (03048) (03049) (03050)
Y11:	00002 (06232)	(03036) (03037)	(02155) (06228) (06343)
Y01:	00002 (03783)	(02049) (02091)	(06313) (06314) (06319) (06324)
Y2:	0000F (02043)	(02201) (04002)	
ZFR0:	0070A (00107)	(06325) (06326) (06392)	

LINES WITH ERRORS: 0 (MAP VERSION 000101.10) E- 0

CSPT PATCHES TO MAP-300 SHAPII (REL 3.5) EXEC
 PROBLEM REPORT #1201) NEWSLETTER #23, PAGE 55
 PROBLEM REPORT #1203) NEWSLETTER #23, PAGE 59
 PROBLEM REPORT #3201) NEWSLETTER #23, PAGE 72
 PROBLEM REPORT #3202) NEWSLETTER #24, PAGE 71
 PROBLEM REPORT #0001) NEWSLETTER #--, PAGE --

PAGE 2
 PAGE 3
 PAGE 4
 PAGE 5
 PAGE 6
 PAGE 7

PAGE 1: (00001) (00016) (00016P) (MS0.1, 10-Nov-00 19:15:52, Ed: KFIELD

(00001) (00001) (00016) (00016P) (MS0.1, 10-Nov-00 19:15:52, Ed: KFIELD

(00002) * CSPI PATCHES TO MAP-300 SWAPII (REL 3.5) EXEC
 (00003) *
 (00004) * THIS PATCH FILE ASSUMES MAP HAS BEEN LOADED WITH
 (00005) * SWAPII EXEC, RELEASE 3.5.
 (00006) *
 (00007) * THIS PATCH ASSUMES REV. 10 (OR LATER) MICROCODE.
 (00008) *
 (00009) * DEFINE 'SAP' (KEY ADDRESS FIELD) AND 'LAP' (LOAD
 (00010) * ADDRESS FIELD) INSTRUCTIONS TO ASSEMBLER.
 (00011) * THESE INSTRUCTIONS ARE NEW CSPI INSTRUCTIONS
 (00012) * AVAILABLE WITH 'REV 10' CSPI MICROCODE.
 (00013) * THEY SET/LOAD THE LOW ORDER 17 BITS OF THE
 (00014) * REFERENCED FULL-WORD FROM/INTO THE INDICATED
 (00015) * REGISTER.
 (00016) *
 (00017) * OPADD SAP,(1 -LS. 14) + (29 -LS. 0) + \$E7
 (00018) * OPADD LAP,(1 -LS. 14) + (29 -LS. 0) + \$E8
 (00019) *
 (00020) * DEFINE 00 TO BE CURRENT TOP OF PATCH AREA
 (00021) * (UPDATE AS PATCH SPACE IS USED).
 (00022) * (SPACE FROM \$1C78-\$1C97 IS USED BY REL. 3.5)
 (00023) *
 (00024) * 00 = \$1C98
 (00025) *
 (00026) * DEFINE REL. 3.5 SYMBOLS
 (00027) *
 (00028) * AP\$WORD0 = \$0F63
 (00029) * BCT\$0A = \$0502
 (00030) * BCT\$AD = \$0604
 (00031) * AP\$BCL = \$0240
 (00032) * AP\$CSL = \$024A
 (00033) * SCD\$04H = \$0E30
 (00034) * W\$ = 1
 (00035) *
 00001C98
 00000F63
 00000502
 00000604
 00000240
 0000024A
 00000E30
 00000001

PAGE 3: (00ND3)C0A10>00N16P.N50.1, 10-NOV-80 19:15:52, EDI KFIELD
 PROBLEM REPORT 01201: NEWSLETTER 023, PAGE 55

(00036) *PROBLEM REPORT 01201: NEWSLETTER 023, PAGE 55
 (00037) *
 (00038) *
 (00039) * PATCH PREBINDING ROUTINES TO WORK WITH PREBINDING
 (00040) * BUFFERS ABOVE \$7FFF (16K).
 (00041) * (NOTE: THIS PATCH USES PATCH SPACE.)
 (00042) *

00000260 (00043) BL = \$0E60
 00E60 023A0502 (00044) LAF R3, DCT\$0A(R5)

0000076C (00045) *
 0076C 02440507 (00046) BL = \$0F6C
 (00047) LAF R4, DCT\$0A(R2)

0000070C (00048) *
 007AC 07540502 (00049) BL = \$0F8C
 (00050) SAF R5, DCT\$0A(R2)

00000701 (00051) *
 007A1 00001C90 (00052) BL = \$0F01
 (00053) JMP P12010
 (00054) P12010:

00000E50 (00055) *
 00E50 00001C00 (00056) BL = \$0E50
 (00057) JMP P1201C

00001C90 (00058) *
 00001C90 (00059) BL = 00
 (00060) P12010:

01C90 C060024A (00061) MOVWRL R6, AP\$CSL
 01C9A 9160FFFF (00062) ANDIR R6, \$FFFF
 01C9C 9A70FFFF (00063) ANDIR R7, \$FFFF
 01C9E 00000703 (00064) JMP P1201A

00000E50 (00065) *
 01CA0 9A30FFFF (00066) P1201C:
 (00067) ANDIR R3, \$FFFF
 01CA2 00000E30 (00068) JMP SCD\$04N
 (00069) *

00001C14 (00070) *
 00001C14 (00071) *
 (00072) *
 00 = BL

PAGE 4: (RNDJ)DCAL16>RNDJSP-MSD.1, 18-NOV-88 19115152, Ed: KFIELD
 PROBLEM REPORT #1203, NEWSLETTER #23, PAGE 59

(00073) *PROBLEM REPORT #1203, NEWSLETTER #23, PAGE 59
 (00074) *
 (00075) * PATCH STANDARD BINDING Routines TO CORRECTLY BIND
 (00076) * ARRAY FUNCTIONS RESIDING ABOVE 16K ON BUS 1.
 (00077) * (NOTE: THIS PATCH USES PATCH SPACE.)
 (00078) *

0000F1E (00079) HL = \$0F1E
 00F1E 00001CA4 (00080) JMP P1203B
 (00081) P1203A:
 (00082) *

00001CA4 (00083) HL = 10
 (00084) P1203B:
 (00085) ANDIR R6,\$FFFF

01CA4 916FFFFF (00086) INCR R6,HS
 01CA6 2661 (00087) INCR R7,HS
 01CA7 2671 (00088) JMP P1203A
 01CA8 0000FF20 (00089) *

(00090) *
 00001CA4 (00091) HL = HL
 (00092) *

PAGE 5: (RNDJ)DCAL16>RNDJSP-MSD.1, 18-NOV-88 19115152, Ed: KFIELD
 PROBLEM REPORT #3201, NEWSLETTER #23, PAGE 72

(00093) *PROBLEM REPORT #3201, NEWSLETTER #23, PAGE 72
 (00094) *
 (00095) * PATCH FFT SPECIAL BINDING Routines TO ALLOW FOR
 (00096) * CORRECT PRE-BINDING OF FFTM.
 (00097) *

0005534 (00098) HL = \$5534
 05514 0000FF63 (00099) JMP AP\$BNDDB
 (00100) *

(00101) *

PAGE 6: (000030CA16)00016P.MSD.1, 18-NOV-88 19:15:52, Ed: KFIELD
 PROBLEM REPORT 03202) NEWSLETTER 024, PAGE 71

(00102) *PROBLEM REPORT 03202) NEWSLETTER 024, PAGE 71
 (00103) *
 (00104) * PATCH TO MAKE PPTM PREDIABLE WITH PREDIABLE BUFFER
 (00105) * ABOVE \$7FFF.
 (00106) * (NOTE: THIS PATCH USES PATCH SPACE.)
 (00107) *
 (00108) *
 (00109) *
 (00110) *

000054A6 (00111) BL = \$54A6
 054A6 00001CA2 (00112) JMP P32020
 (00113) P32020
 (00114) *

00001CA2 (00115) BL = 00
 01CAA F0300240 (00116) P32020: MOVNR R3,AP\$0SL
 01CAC 9A30FF0F (00117) ANDIR R3,\$0FFF
 01CAE 000054A0 (00118) JMP P3202A
 (00119) *
 (00120) *

00001C00 (00121) 00 = 0L
 (00122) *

PAGE 7: (000030CA16)00016P.MSD.1, 18-NOV-88 19:15:52, Ed: KFIELD
 PROBLEM REPORT 03202) NEWSLETTER 024, PAGE 71

(00123) *PROBLEM REPORT 03202) NEWSLETTER 024, PAGE 71
 (00124) *
 (00125) * PATCH TO EXEC UTILITY "VALBUFS" TO ALLOW FOR
 (00126) * CORRECT VALIDATION OF BUFFERS WITH NEGATIVE
 (00127) * SAMPLE INCREMENTS (SI'S).
 (00128) * KFIELD 7/88
 (00129) *

00001C5E (00130) BL = \$1C5E
 01C5E F0340604 (00131) MOVNR R3,BCT\$AD(R2) LOAD VECTOR SI
 (00132) *

00001C62 (00133) BL = \$1C62
 01C62 C2340605 (00134) MULNR R3,BCT\$AD+H\$(R2) SHULT BY SIZE-1
 (00135) *

01C64 (00136) END

AP3NDR01	00767	(00620)	(00099)
AP50SL1	00740	(00031)	(00117)
AP53SL1	00744	(00032)	(00061)
OC13A01	00604	(00030)	(00131)
OC13A01	00562	(00029)	(00044)
MS1	00091	(00034)	(00006)
P1201A1	00003	(00034)	(00064)
P1201B1	01C90	(00053)	(00060)
P1201C1	01C10	(00057)	(00066)
P1201A1	00F20	(00001)	(00000)
P1203B1	01C14	(00000)	(00004)
P3202A1	05400	(00113)	(00119)
P3202B1	01C1A	(00112)	(00116)
SCD504M1	00F30	(00033)	(00060)

LINES WITH PARAGS: 8 (MAP VERSION 00101.10) 8- 6

PAGE 1: (0000) <0C116>00N16T.NSU.1, 10-NOV-00 19115102, Ed: KFIELD
 (00001) J(0000) <0C116>00N16T.NSU.1, 10-NOV-00 19115102, Ed: KFIELD

09850	F4127640 (00055)	DATA --90681341
09852	F2168DC0 (00056)	DATA --89571356
09854	F11258C0 (00057)	DATA --80337240
09856	F5162C0 (00058)	DATA --86967121
09858	E05PC1C0 (00059)	DATA --85448478
0985A	E0392D40 (00060)	DATA --83768241
0985C	E0939440 (00061)	DATA --81912912
0985E	E63B6440 (00062)	DATA --79868747
09860	E35B24C0 (00063)	DATA --77621973
09862	E0342840 (00064)	DATA --75159476
09864	DCC207C0 (00065)	DATA --72467132
09866	D908F840 (00066)	DATA --69534210
09868	D4ED8240 (00067)	DATA --66349818
0986A	D084D040 (00068)	DATA --62985408
0986C	C8C4F040 (00069)	DATA --59194915
0986E	C6ACF340 (00070)	DATA --55215304
09870	C13CE7C0 (00071)	DATA --50967117
09872	B0765D40 (00072)	DATA --46454968
09874	B5C5140 (00073)	DATA --41687981
09876	AFF354C0 (00074)	DATA --36680087
09878	A8419440 (00075)	DATA --31450189
0987A	A14EE40 (00076)	DATA --26022125
0987C	9124AE40 (00077)	DATA --20424442
0987E	92C09440 (00078)	DATA --14689926
09880	8D595C0 (00079)	DATA --88054935
09882	8C97378F (00080)	DATA --82958537
09884	3C97413F (00081)	DATA --808054942
09886	8A559640 (00082)	DATA --814689934
09888	12C098C0 (00083)	DATA --82042450
0988A	1124AF40 (00084)	DATA --826022132
0988C	214EEC0 (00085)	DATA --831450195
0988E	204199C0 (00086)	DATA --83668895
09890	2FP35540 (00087)	DATA --841687988
09892	355C51C0 (00088)	DATA --846454975
09894	3B765DC0 (00089)	DATA --850967123
09896	413CE840 (00090)	DATA --855215311
09898	46ACF3C0 (00091)	DATA --859194921
0989A	4BC4FDC0 (00092)	DATA --862905414
0989C	5084D0C0 (00093)	DATA --866349822
0989E	54E08240 (00094)	DATA --869534215
098A0	5980F8C0 (00095)	DATA --872467136
098A2	5CC207C0 (00096)	DATA --875159479
098A4	60342040 (00097)	DATA --877621976
098A6	635B2040 (00098)	DATA --879868749
098A8	663B6440 (00099)	DATA --881912915
098AA	68093940 (00100)	DATA --883768244
098AC	6A392DC0 (00101)	DATA --885448480
098AE	6D5FC1C0 (00102)	DATA --73101083
098B0	DD91C340 (00103)	DATA --69104202
098B2	D0741040 (00104)	DATA --64634765
098B4	D28B04C0 (00105)	DATA --59672950
098B6	CC61A240 (00106)	DATA --54208627
098B8	C5631540 (00107)	

CODING TABLE FOR K2 (32)

CTHK2:

0908A	0DC08DC0 (00100)	DATA --40243076
0908C	057F05C0 (00109)	DATA --41795417
0908E	AC8A0740 (00110)	DATA --34896366
0908B	A3510640 (00111)	DATA --27597119
0908D	998F2540 (00112)	DATA --19965846
0908F	0E7749C0 (00113)	DATA --12002789
0908G	D208503F (00114)	DATA --04045165
0908H	5208443F (00115)	DATA 0.04045156
0908A	0E774940 (00116)	DATA 0.12082781
0908C	190E24C0 (00117)	DATA 0.19965837
0908E	27530540 (00118)	DATA 0.27597111
0908B	2CAAD6C0 (00119)	DATA 0.34896358
0908D	357F84C0 (00120)	DATA 0.41795409
0908F	3DC08CC0 (00121)	DATA 0.48243069
0908G	456314C0 (00122)	DATA 0.54200621
0908H	4C61A1C0 (00123)	DATA 0.59672952
0908A	520804C0 (00124)	DATA 0.64634768
0908C	50741040 (00125)	DATA 0.69104198
0908E	5D91C2C0 (00126)	DATA 0.73101079
0908B	621060C0 (00127)	DATA 0.76652206
0908D	602163C0 (00128)	DATA 0.79709401
0908F	6A00E240 (00129)	DATA 0.82546644
0908G	6CBF6DC0 (00130)	DATA 0.84959197
0908H	6F707140 (00131)	DATA 0.87061901
0908A	71C68BC0 (00132)	DATA 0.00800300
0908C	73CD37C0 (00133)	DATA 0.90470025
0908E	750CECC0 (00134)	DATA 0.91036321
0908B	DCC263C0 (00135)	DATA --72460231
0908D	D4A54FC0 (00136)	DATA --06129490
0908F	C81C9640 (00137)	DATA --58608991
0908G	C81C6240 (00138)	DATA --50006620
0908H	3AE0240 (00139)	DATA --40376312
0908A	ASF78940 (00140)	DATA --29662241
0908C	9739EF40 (00141)	DATA --10145542
0908E	FDIAC80F (00142)	DATA --06100426
0908B	7DIAC80F (00143)	DATA 0.06100626
0908D	1739EE40 (00144)	DATA 0.10145541
0908F	25F78940 (00145)	DATA 0.29662241
0908G	33AF0240 (00146)	DATA 0.40376112
0908H	401C6240 (00147)	DATA 0.50006619
0908A	401C9640 (00148)	DATA 0.58608990
0908C	54A54F40 (00149)	DATA 0.66129489
0908E	5CC263C0 (00150)	DATA 0.72460230
0908B	C4292040 (00151)	DATA --53250506
0908D	9983C0C0 (00152)	DATA --45079015
0908F	AE20A940 (00153)	DATA --36076742
0908G	A1096740 (00154)	DATA --26347055
0908H	940A40C0 (00155)	DATA --16050062
0908A	EF6AB00F (00156)	DATA --05391451
0908C	626AB00F (00157)	DATA 0.05391451
0908E	140A40C0 (00158)	DATA 0.16050061
0908B	21B066C0 (00159)	DATA 0.26347053
0908D	2F20A040 (00160)	DATA 0.36076741

DECODING TABLE FOR K3 (16)

DECODING TABLE FOR K4 (16)

09924	3987C0C0 (00161)	DATA	0.45879814
09926	44292040 (00162)	DATA	0.53250505
09928	4D7C8640 (00163)	DATA	0.68532677
0992A	55AC2340 (00164)	DATA	0.66931576
0992C	5CC267C0 (00165)	DATA	0.72460230
0992E	62D25C40 (00166)	DATA	0.77200470
09930	8C9966C0 (00167)	DATA	0.77343143
09932	82C097C0 (00168)	DATA	0.39772316
09934	A078A340 (00169)	DATA	0.31618157
09936	9D664CC0 (00170)	DATA	0.22968443
09938	91D7D040 (00171)	DATA	0.13940015
0993A	DFB78F3F (00172)	DATA	0.04673719
0993C	5F878CB7 (00173)	DATA	0.04673717
0993E	11D7D040 (00174)	DATA	0.13940014
09940	1D664CC0 (00175)	DATA	0.22968442
09942	2078A340 (00176)	DATA	0.31618155
09944	32E897C0 (00177)	DATA	0.39772316
09946	3C9966C0 (00178)	DATA	0.47343142
09948	45786240 (00179)	DATA	0.54273631
0994A	4D7C8640 (00180)	DATA	0.68532677
0994C	54A54F40 (00181)	DATA	0.66129489
0994E	5AF92440 (00182)	DATA	0.71872024
09950	A9578440 (00183)	DATA	0.32290331
09952	9E9E0BC0 (00184)	DATA	0.23400363
09954	924022C0 (00185)	DATA	0.14250225
09956	E1E0DF8F (00186)	DATA	0.04781699
09958	61E0D0BF (00187)	DATA	0.47801697
0995A	124022C0 (00188)	DATA	0.14250224
0995C	1E0F0840 (00189)	DATA	0.23400362
0995E	29578440 (00190)	DATA	0.32290330
09960	31F43840 (00191)	DATA	0.48589048
09962	3DC65740 (00192)	DATA	0.40261537
09964	46001BC0 (00193)	DATA	0.55258511
09966	4E2A1640 (00194)	DATA	0.61554603
09968	55F46A40 (00195)	DATA	0.67152148
0996A	5C418940 (00196)	DATA	0.72075575
0996C	618F6940 (00197)	DATA	0.76365393
0996E	667F294E (00198)	DATA	0.80072517
09970	41F8908A (00199)	DATA	0.000000307255794
09972	5085E13A (00200)	DATA	0.000000375037437
09974	6209C33A (00201)	DATA	0.000000459727080
09976	78C3003A (00202)	DATA	0.000000562341396
09978	893R7780 (00203)	DATA	0.000000687059929
0997A	80480330 (00204)	DATA	0.000000841395202
0997C	8D094E39 (00205)	DATA	0.000001029200590
0997E	10E5A200 (00206)	DATA	0.000001258925660
09980	14825000 (00207)	DATA	0.000001539926070
09982	19402830 (00208)	DATA	0.000001836493300
09984	1ECC0000 (00209)	DATA	0.000002304093330
09986	2503E430 (00210)	DATA	0.000002818303390
09988	2E456700 (00211)	DATA	0.000003447466740
0998A	30996230 (00212)	DATA	0.000004216965940
0998C	30996230 (00213)	DATA	0.000004216965940

JCODING TABLE FOR K5 (16)

JCODING TABLE FOR K6 (16)

JCODING TABLE FOR (OVERALL) ENERGY (64)

0990C	4530043R (00214)	DATA	0.0000005150222220
0990E	54A7000W (00215)	DATA	0.0000006309575000
0990H	6790000B (00216)	DATA	0.0000007717916330
0990J	7E05003B (00217)	DATA	0.000000940610360
0990L	89A7003C (00218)	DATA	0.0000011547022100
0990N	99D9600C (00219)	DATA	0.0000014125376900
0990P	0E7F7A0C (00220)	DATA	0.000001720263900
0990R	110A073C (00221)	DATA	0.0000021134093600
0990T	15AFC03C (00222)	DATA	0.0000025052352200
0990V	1A06F00C (00223)	DATA	0.0000031622702000
0990X	2M72090C (00224)	DATA	0.0000038601200700
0990Z	2700D93C (00225)	DATA	0.0000047315136300
0990A	300C010C (00226)	DATA	0.0000057076205800
0990B	3B63050C (00227)	DATA	0.0000070794509000
0990C	40A4710C (00228)	DATA	0.000008659644500
0990E	50D8003C (00229)	DATA	0.0000105925379000
0990G	6C00A03C (00230)	DATA	0.0000129560696000
0990I	004F3530 (00231)	DATA	0.0000158409336000
0990K	0A200430 (00232)	DATA	0.0000193065209000
0990M	0C6EC00D (00233)	DATA	0.0000237137406000
0990O	0F35A30 (00234)	DATA	0.0000290060133000
0990Q	179A390B (00235)	DATA	0.0000354813456000
0990S	16C1303D (00236)	DATA	0.0000434010303000
0990U	10D5693D (00237)	DATA	0.0000530004504000
0990W	2200A30 (00238)	DATA	0.0000649301709000
0990Y	29A5003D (00239)	DATA	0.0000794320262000
0990A	32F0F00D (00240)	DATA	0.0000971628124000
0990C	3F4FC00D (00241)	DATA	0.0001108502310000
0990E	4C305D3D (00242)	DATA	0.0001453704520000
0990G	5030A30 (00243)	DATA	0.0001770279600000
0990I	7200100D (00244)	DATA	0.0002175204340000
0990K	0007FA3E (00245)	DATA	0.0002660725400000
0990M	0AA2700E (00246)	DATA	0.0003254617900000
0990O	0009000E (00247)	DATA	0.0003901072000000
0990Q	07F4FA0E (00248)	DATA	0.0004069675720000
0990S	1304C60E (00249)	DATA	0.0005956622120000
0990U	17E0170E (00250)	DATA	0.0007206182370000
0990W	10345A0E (00251)	DATA	0.0008912509730000
0990Y	2309213E (00252)	DATA	0.0010901045700000
0990A	2002630E (00253)	DATA	0.0013335215500000
0990C	3573450E (00254)	DATA	0.0016111730900000
0990E	4161790E (00255)	DATA	0.0019526254000000
0990G	47F9053E (00256)	DATA	0.0024406191100000
0990I	6103343E (00257)	DATA	0.0029853027400000
0990K	77A9060E (00258)	DATA	0.0036517415100000
0990M	0925E00F (00259)	DATA	0.0044668363400000
0990O	0030A30F (00260)	DATA	0.0054630661100000
0990Q	0000C3F (00261)	DATA	0.0066034391900000
0990S	10072C0F (00262)	DATA	0.0081752307000000
0990U	147A113F (00263)	DATA	0.0099999999800000
0990W	370FC04A (00264)	DATA	0.43651583
0990Y	721402C0 (00265)	DATA	0.09125094
0990A	0040541 (00266)	DATA	1.65958609

JCODING TABLE FOR DELTA-GAIN (4)

09976 1F400043 (00267)	DATA 1000.00000000		
09970 7034C740 (00268)	DATA -0.97017319		)/CODING TABLE FOR RESIDUAL (4)
09974 00000000 (00269)	DATA 0.00000000		
0997C 7034C740 (00270)	DATA 0.97017319		
0997E 1F400043 (00271)	DATA 1000.00000000		
09972 (00272)			
09973 (00273)			
09974 (00274)			
09975 (00275)			
09976 (00276)			
09977 (00277)			
09978 (00278)			
09979 (00279)			
09980 (00280)			
09981 (00281)			
09982 (00282)			
09983 (00283)			
09984 (00284)			
09985 (00285)			
09986 (00286)			
09987 (00287)			
09988 (00288)			
09989 (00289)			
09990 (00290)			
09991 (00291)			
09992 (00292)			
09993 (00293)			
09994 (00294)			
09995 (00295)			
09996 (00296)			
09997 (00297)			
09998 (00298)			
09999 (00299)			
09A00 (00300)			
09A01 (00301)			
09A02 (00302)			
09A03 (00303)			
09A04 (00304)			
09A05 (00305)			
09A06 (00306)			
09A07 (00307)			
09A08 (00308)			
09A09 (00309)			
09A10 (00310)			
09A11 (00311)			
09A12 (00312)			
09A13 (00313)			
09A14 (00314)			
09A15 (00315)			
09A16 (00316)			
09A17 (00317)			
09A18 (00318)			
09A19 (00319)			

09A20 0E76C8C0 (00277)	DVLC3:	DATA --.40000000	
09A21 0E0916C0 (00278)	DVLC3:	DATA --.36600000	
09A22 9F306440 (00279)	DVLC3:	DATA --.24400000	
09A23 0F900240 (00280)	DVLC3:	DATA --.12200000	
09A24 00000000 (00281)	DVLC3:	DATA --.00000000	
09A25 0F900240 (00282)	DVLC3:	DATA --.12200000	
09A26 1F306440 (00283)	DVLC3:	DATA --.24400000	
09A27 2E0916C0 (00284)	DVLC3:	DATA --.36600000	
09A28 7F5100C0 (00285)	DVLC3:	DATA --.91656251	
09A29 0CC20F40 (00286)	DVLC3:	DATA --.04968750	
09A30 2433340 (00287)	DVLC3:	DATA --.78201250	
09A31 00A30740 (00288)	DVLC3:	DATA --.71593750	
09A32 03147AC0 (00289)	DVLC3:	DATA --.64906249	
09A33 0A051EC0 (00290)	DVLC3:	DATA --.58210749	
09A34 01F5C240 (00291)	DVLC3:	DATA --.51531249	
09A35 0966640 (00292)	DVLC3:	DATA --.44043748	
09A36 00D70A40 (00293)	DVLC3:	DATA --.38156248	
09A37 0047A0C0 (00294)	DVLC3:	DATA --.31468748	
09A38 9F0051C0 (00295)	DVLC3:	DATA --.24701249	
09A39 9720F5C0 (00296)	DVLC3:	DATA --.18093749	
09A40 06999C0 (00297)	DVLC3:	DATA --.11406249	
09A41 20A3050F (00298)	DVLC3:	DATA --.04710749	
09A42 2051EE3F (00299)	DVLC3:	DATA --.01968752	
09A43 00107040 (00300)	DVLC3:	DATA --.00656252	
09A44 (00301)			
09A45 (00302)			
09A46 (00303)			
09A47 (00304)			
09A48 (00305)			
09A49 (00306)			
09A50 (00307)			
09A51 (00308)			
09A52 (00309)			
09A53 (00310)			
09A54 (00311)			
09A55 (00312)			
09A56 (00313)			
09A57 (00314)			
09A58 (00315)			
09A59 (00316)			
09A60 (00317)			
09A61 (00318)			
09A62 (00319)			

09A30 FE3CE9C0 (00302)	DVLC1:	DATA --.98623391	
09A31 FE04A40 (00303)	DVLC1:	DATA --.98451736	
09A32 F0C576C0 (00304)	DVLC1:	DATA --.98250064	
09A33 F07E7040 (00305)	DVLC1:	DATA --.98042203	
09A34 F02E0C40 (00306)	DVLC1:	DATA --.97790879	
09A35 F0D537C0 (00307)	DVLC1:	DATA --.97525092	
09A36 FC700F40 (00308)	DVLC1:	DATA --.97219076	
09A37 FC000540 (00309)	DVLC1:	DATA --.96875066	
09A38 F0019940 (00310)	DVLC1:	DATA --.96489257	
09A39 F1F3E1C0 (00311)	DVLC1:	DATA --.96056771	
09A40 F15519C0 (00312)	DVLC1:	DATA --.95572206	
09A41 F9A34D40 (00313)	DVLC1:	DATA --.95029607	
09A42 F0DC55C0 (00314)	DVLC1:	DATA --.94422411	
09A43 F7F007C0 (00315)	DVLC1:	DATA --.93743417	
09A44 F70530C0 (00316)	DVLC1:	DATA --.92984749	
09A45 F5E00940 (00317)	DVLC1:	DATA --.92137019	
09A46 F40A38C0 (00318)	DVLC1:	DATA --.91193306	
09A47 F16173C0 (00319)	DVLC1:	DATA --.90141150	

DECODING TABLE FOR K2 (32)

DVLK2:

09ABE B1260040 (00373)	DATA --38399474
09AC0 A00*1240 (00374)	DATA --31292941
09AC2 9F7C1C0 (00375)	DATA --23017049
09AC4 940R19C0 (00376)	DATA --16049504
09AC6 0A5A04C0 (00377)	DATA --00077100
09AC8 00000000 (00378)	DATA 0.00000000
09ACA 0A500440 (00379)	DATA 0.00077099
09ACC 14001940 (00380)	DATA 0.16049496
09ACE 1E7C1140 (00381)	DATA 0.23017041
09AD0 200F1140 (00382)	DATA 0.31292932
09AD2 31260C00 (00383)	DATA 0.38399466
09AD4 39034C00 (00384)	DATA 0.45070433
09AD6 41A61740 (00385)	DATA 0.51200119
09AD8 40F70040 (00386)	DATA 0.57003706
09ADA 4FA300C0 (00387)	DATA 0.62216294
09ADC 55A090C0 (00388)	DATA 0.66929903
09ADE 5015C340 (00389)	DATA 0.71160164
09AE0 5F0939C0 (00390)	DATA 0.74930502
09AE2 6427A0C0 (00391)	DATA 0.70270507
09AE4 67F3E440 (00392)	DATA 0.81213313
09AE6 604100C0 (00393)	DATA 0.83793800
09AE8 6E23E0C0 (00394)	DATA 0.86047144
09AFA 70A655C0 (00395)	DATA 0.88007615
09AFC 72037940 (00396)	DATA 0.89707066
09AEE 740551C0 (00397)	DATA 0.91170370
09AF0 E0590400 (00398)	DATA --75245064
09AF2 000F0E40 (00399)	DATA --69433287
09AF4 000F9340 (00400)	DATA --62547535
09AF6 C5C06340 (00401)	DATA --54526930
09AF8 0A110940 (00402)	DATA --45360014
09AFA AC00C0C0 (00403)	DATA --35134251
09AFC 9E047A40 (00404)	DATA --23900272
09AFE 0F947740 (00405)	DATA --12171034
09000 00000000 (00406)	DATA 0.00000000
09002 0F947740 (00407)	DATA 0.12171033
09004 1E047A40 (00408)	DATA 0.23900271
09006 2CF0C040 (00409)	DATA 0.35134250
09008 3A1100C0 (00410)	DATA 0.45360013
0900A 45C062C0 (00411)	DATA 0.54526937
0900C 500F9340 (00412)	DATA 0.62547535
0900E 500F0E40 (00413)	DATA 0.69433286
09010 C0F70340 (00414)	DATA --57005347
09012 0F1200C0 (00415)	DATA --49273609
09014 04110A40 (00416)	DATA --40677000
09016 A00F0940 (00417)	DATA --31293977
09018 903510C0 (00418)	DATA --21257565
0901A 0DC319C0 (00419)	DATA --10751649
0901C 00000000 (00420)	DATA 0.00000000
0901E 0DC319C0 (00421)	DATA 0.10751640
09020 103510C0 (00422)	DATA 0.21257564
09022 200F0940 (00423)	DATA 0.31293977
09024 34110A40 (00424)	DATA 0.40677000
09026 3F1200C0 (00425)	DATA 0.49273609

DECODING TABLE FOR K3 (16)

DECODING TABLE FOR K4 (16)

PAGE 10: (ABNDJ)CUCALG>NBI6T.MSD.1, 10-NOV-80 19:15:02, ED: KPIZLD
CODING AND DECODING TABLES

09820 40F70340 (00426) DATA 0-57005347
09821 510065C0 (00427) DATA 0-63043908
09822 59596740 (00428) DATA 0-69080000
09823 5FE90940 (00429) DATA 0-74932021
09824 C1232640 (00430) DATA --50090009
09825 070A4040 (00431) DATA --43634920
09826 40C6F240 (00432) DATA --35763304
09827 A3019540 (00433) DATA --27340579
09828 97A00DC0 (00434) DATA --10493246
09829 00F44C0 (00435) DATA --09327063
09830 00000000 (00436) DATA 0-00000000
09831 00F040C0 (00437) DATA 0-09327062
09832 17A00DC0 (00438) DATA 0-10493246
09833 23019540 (00439) DATA 0-27340579
09834 20C6F1C0 (00440) DATA 0-35763303
09835 370A4040 (00441) DATA 0-43634920
09836 4123E640 (00442) DATA 0-50890000
09837 499605C0 (00443) DATA 0-57489004
09838 512C13C0 (00444) DATA 0-63415766
09839 57E93940 (00445) DATA 0-60600496
09840 AEB09640 (00446) DATA --36516072
09841 A3C616C0 (00447) DATA --27940269
09842 9034C8C0 (00448) DATA --10910990
09843 8C3695C0 (00449) DATA --09541500
09844 00000000 (00450) DATA 0-00000000
09845 0C3695C0 (00451) DATA 0-09541579
09846 1034C0C0 (00452) DATA 0-10910909
09847 23C616C0 (00453) DATA 0-27940267
09848 2E009640 (00454) DATA 0-36516071
09849 30F00040 (00455) DATA 0-44506935
09850 425D2540 (00456) DATA 0-51046750
09851 40F0340 (00457) DATA 0-58494609
09852 527092C0 (00458) DATA 0-64439614
09853 5935F3C0 (00459) DATA 0-69695902
09854 501909C0 (00460) DATA 0-74296685
09855 64350640 (00461) DATA 0-70200344
09856 64350640 (00462) DATA 0-70200344

DECODING TABLE FOR K5 (16)

DECODING TABLE FOR K6 (16)

DECODING TABLE FOR SQR(ENERGY) (64)

DVLC:

09870 5762F900 (00464) DATA 0-0001666767600000
09871 60A60130 (00465) DATA 0-0001843423110000
09872 6AE45530 (00466) DATA 0-0002030001790000
09873 76309930 (00467) DATA 0-0002254000310000
09874 002C030E (00468) DATA 0-0002493076930000
09875 0000000E (00469) DATA 0-0002750195510000
09876 09FF070E (00470) DATA 0-0003050520070000
09877 000F200E (00471) DATA 0-0003373044230000
09878 0C3A250E (00472) DATA 0-0003731427710000
09879 0005E73E (00473) DATA 0-0004126910390000
09880 0FF4010E (00474) DATA 0-0004564309060000
09881 100A900E (00475) DATA 0-0005000666310000
09882 1240700E (00476) DATA 0-0005503095120000
09883 1430D20E (00477) DATA 0-0006174030700000
09884 1660013E (00478) DATA 0-0006082920105000

0900E 1007FF3E (00479)	DATA	0.000755309580000
0900F 105070E (00480)	DATA	0.000835362631000
0900G 1F4633E (00481)	DATA	0.000923900121000
0900H 217P100E (00482)	DATA	0.001021021530000
0900I 250020E (00483)	DATA	0.001130121220000
0900J 207421E (00484)	DATA	0.001249899330000
0900K 204C300E (00485)	DATA	0.001302372370000
0900L 321930E (00486)	DATA	0.001520005700000
0900M 376070E (00487)	DATA	0.001690927750000
0900N 3047E73E (00488)	DATA	0.001870143910000
0900O 43C6900E (00489)	DATA	0.002060354700000
0900P 4AF50E3E (00490)	DATA	0.002287573410000
0900Q 52E7650E (00491)	DATA	0.002530026190000
0900R 5000C03E (00492)	DATA	0.002798176140000
0900S 65609C3E (00493)	DATA	0.003094746890000
0900T 7020100E (00494)	DATA	0.003322748020000
0900U 7C0R348E (00495)	DATA	0.003705515550000
0900V 0097003F (00496)	DATA	0.004106730540000
0900W 0070033F (00497)	DATA	0.004638469660000
0900X 0A7D013F (00498)	DATA	0.005121230760000
0900Y 0099433F (00499)	DATA	0.005640232200000
0900Z 0CD4500F (00500)	DATA	0.006264335030000
0901A 0E34693F (00501)	DATA	0.006920273240000
0901B 0FB1063F (00502)	DATA	0.007662508290000
0901C 1150300F (00503)	DATA	0.008474713420000
0901D 1321C3F (00504)	DATA	0.009372923060000
0901E 153AF13F (00505)	DATA	0.010366329900000
0901F 177AF90F (00506)	DATA	0.011465025700000
0901G 19F00F3F (00507)	DATA	0.012601600000000
0901H 1C0RAA0F (00508)	DATA	0.014024100500000
0901I 1FC3F43F (00509)	DATA	0.015514472100000
0901J 2321D50F (00510)	DATA	0.017154300000000
0901K 260R100F (00511)	DATA	0.018972520500000
0901L 2AF9520F (00512)	DATA	0.020983359500000
0901M 2F07510F (00513)	DATA	0.023207321100000
0901N 3490E50F (00514)	DATA	0.025666994300000
0901O 3A2327E (00515)	DATA	0.028307361000000
0901P 404C920F (00516)	DATA	0.031390051400000
0901Q 4710200F (00517)	DATA	0.034723622700000
0901R 4FA6003F (00518)	DATA	0.038403871500000
0901S 56FC030F (00519)	DATA	0.042474189000000
0901T 6034E53F (00520)	DATA	0.046975890200000
0901U 6A673C0F (00521)	DATA	0.051954722000000
0901V 75AF303F (00522)	DATA	0.057461244100000
0901W 082273C0 (00523)	DATA	0.063551381000000
0901X 00FF2A40 (00524)	DATA	0.070287000400000
0901Y 09F34540 (00525)	DATA	0.077736505300000
0901Z 00013740 (00526)	DATA	0.085975561300000
0902A 0C2006C0 (00527)	DATA	0.095087840000000
0902B 3F011240 (00528)	DATA	0.40977802
0902C 65AC0C40 (00529)	DATA	0.79432024
0902D 00F9E4C1 (00530)	DATA	1.12201844
0902E 00F0A41 (00531)	DATA	1.49623564

DECODING TABLE FOR SORT(DELTA-GAIN) (4)

PAGE 121 (00002)DC116>000167-M30.1, 10-Nov-88 19:15:02, Ed: KFIELD
CODING AND DECODING TABLES

09070 0C000541 (00532) DVL0:	DATA -1.50409251	)DECODING TABLE FOR REGIONAL (4) (CODED 1,0,2,3)
0907A 09C930C0 (00533)	DATA -0.45145300	
0907C 39C930C0 (00534)	DATA 0.45145300	
0907E 0C000541 (00535)	DATA 1.50409251	
09C00 09C930C0 (00536) DVLUP:	DATA -0.45145300	)DVL0 IN FOLDED BINARY ORDER (4) (CODED 0,1,2,3)
09C02 0C000541 (00537)	DATA -1.50409251	
09C04 39C930C0 (00538)	DATA 0.45145300	
09C06 0C000541 (00539)	DATA 1.50409251	
09C08	END	

PAGE 13: (0000)<00016>000167.MSU.1, 10-NOV-00 19:15:02, EOL: RFIELD
 CNOING AND DECODING TABLES

CTNC1:	0000 (00013)
CTNC2:	0010 (00022)
CTNC3:	0000 (00014)
CTNDC1:	0000 (00264)
CTNG1:	0000 (00199)
CTNR1:	0000 (00039)
CTNR2:	0000 (00103)
CTNR3:	0000 (00135)
CTNR4:	0010 (00151)
CTNR5:	0030 (00167)
CTNR6:	0050 (00183)
CTNR7:	0000 (00260)
DVLC1:	0000 (00276)
DVLC2:	0010 (00285)
DVLC3:	0000 (00277)
DVLC4:	0000 (00520)
DVLC5:	0000 (00463)
DVLC6:	0000 (00302)
DVLC7:	0000 (00366)
DVLC8:	0000 (00390)
DVLC9:	0010 (00414)
DVLC10:	0000 (00430)
DVLC11:	0000 (00446)
DVLC12:	0000 (00532)
DVLC13:	0000 (00536)

LINES WITH ERRORS: 0 (MAP VERSION 000101.10) 2- 0

[000001<00016>000000.MSO.2, 29-Dec-80 15:44:23, Ed: WULF

T A B L E O F C O N T E N T S

BBN 16 ED/S SPEECH CODER INPUT/OUTPUT PROGRAMS	PAGE 2
SYMBOL DEFINITIONS	PAGE 3
PATCHES TO NON-ARRAY AND ARRAY FUNCTION DISPATCH TABLES	PAGE 6
PATCHES TO INTERRUPT SERVICE ROUTINES	PAGE 7
I/O BUFFER DEFINITIONS	PAGE 8
INTEGER SCALAR TABLE DEFINITIONS FOR I/O ROUTINES	PAGE 10
ADAM SINGLE-CHANNEL DOUBLE-BUFFERED D/A OUT	PAGE 12
ADAM INTERFACE AND SYSTEM CLOCKS PROGRAMS	PAGE 15
ADAM SINGLE-CHANNEL DOUBLE-BUFFERED D/A OUT	PAGE 17
ADAM SINGLE-CHANNEL DOUBLE-BUFFERED D/A OUT	PAGE 19
ADAM SINGLE-CHANNEL DOUBLE-BUFFERED D/A OUT	PAGE 21
ADAM SINGLE-CHANNEL DOUBLE-BUFFERED D/A OUT	PAGE 23
ADAM SINGLE-CHANNEL DOUBLE-BUFFERED D/A OUT	PAGE 26
ADAM SINGLE-CHANNEL DOUBLE-BUFFERED D/A OUT	PAGE 29
ADAM SINGLE-CHANNEL DOUBLE-BUFFERED D/A OUT	PAGE 31
ADAM SINGLE-CHANNEL DOUBLE-BUFFERED D/A OUT	PAGE 38
ADAM SINGLE-CHANNEL DOUBLE-BUFFERED D/A OUT	PAGE 45
ADAM SINGLE-CHANNEL DOUBLE-BUFFERED D/A OUT	PAGE 47
ADAM SINGLE-CHANNEL DOUBLE-BUFFERED D/A OUT	PAGE 49

SYMBOL DEFINITIONS

```

(00005) "SYMBOL DEFINITIONS
(00006) ;
00000001 (00007) H$ = 1
00000002 (00008) V$ = 2
00000003 (00009) ;
00000004 (00010) ACQTHR = 10
00000005 (00011) ;
00000006 (00012) ;
00000007 (00013) AFD$ORC = $0E0
00000008 (00014) BITS15 = 0'7777'
00000009 (00015) ;
00000010 (00016) ;BIT-MASKS USED IN THE PROPAR ROUTINE
00000011 (00017) B0 = 1
00000012 (00018) B1 = 2
00000013 (00019) B2 = 4
00000014 (00020) B3 = 8
00000015 (00021) B4 = $10
00000016 (00022) B5 = $20
00000017 (00023) B10 = $1+00
00000018 (00024) B210 = $2+01+00
00000019 (00025) B32 = $3+B2
00000020 (00026) B321 = $3+$2+B1
00000021 (00027) B54 = $5+B4
00000022 (00028) ;BIT-MASKS SHIFTED 1 PLACE LEFT FOR USE IN THE CORPAR ROUTINE
00000023 (00029) C0 = 2
00000024 (00030) C1 = 4
00000025 (00031) C2 = 8
00000026 (00032) C3 = $10
00000027 (00033) C4 = $20
00000028 (00034) C5 = $40
00000029 (00035) C10 = C1+C0
00000030 (00036) C21 = C2+C1
00000031 (00037) C210 = C2+C1+C0
00000032 (00038) C32 = C3+C2
00000033 (00039) C321 = C3+C2+C1
00000034 (00040) C54 = C5+C4
00000035 (00041) ;
000021FC (00042) CSPU$MDS = $21FC
00000036 (00043) ;
00000037 (00044) [C00NDJ]C0CA16>BBN16T-ADDRS.MSD.2, 9-Sep-88 15:08:32, E4: KFIELD
00000038 (00045) ;SYMBOL DEFINITIONS FOR BBN16 CODING/DECODING TABLE ADDRESSES
00000039 (00046) ; TABLES RESIDE ON BUS1, STARTING AT $900.
00000040 (00047) ; TABLE CONTENTS ARE DEFINED IN MODULE "BBN16T.MSD".
00000041 (00048) ;
00000042 (00049) CTHC1=$900
00000043 (00050) CTHC2=CTHC1+ 8*W$
00000044 (00051) CTHC3=CTHC1
00000045 (00052) CTHK1=CTHC2+16*W$
00000046 (00053) CTHK2=CTHK1+64*W$
00000047 (00054) CTHK3=CTHK2+32*W$
00000048 (00055) CTHK4=CTHK3+16*W$
00000049 (00056) CTHK5=CTHK4+16*W$
00000050 (00057) CTHK6=CTHK5+16*W$

```

```

00009970 (00050) CTNC =CTHC+16*W$
00009980 (00059) CTDC=CTHC+64*W$
00009990 (00060) CTNU =CTDC+ 4*W$
00009A00 (00061) DVLC1=CTNU + 4*W$
00009A10 (00062) DVLC2=DVLC1+ 8*W$
00009A20 (00063) DVLC3=DVLC1
00009A30 (00064) DVLC4=DVLC2+16*W$
00009A40 (00065) DVLC5=DVLC1+64*W$
00009A50 (00066) DVLC6=DVLC2+32*W$
00009A60 (00067) DVLC7=DVLC3+16*W$
00009A70 (00068) DVLC8=DVLC4+16*W$
00009A80 (00069) DVLC9=DVLC5+16*W$
00009A90 (00070) DVLC =DVLC6+16*W$
00009B00 (00071) DVLCG=DVLC +64*W$
00009B10 (00072) DVLU =DVLCG+ 4*W$
00009B20 (00073) DVLUF=DVLU + 4*W$
00009C00 (00074)

```

DECODING TABLES

```

00004022 (00075) J$MAP EXEC INTERRUPT ROUTINE ADDRESSES
00004030 (00076) D16$INT1 = $4022 J$MODEM SCROLL INT 1 (J$MODEM)
00004040 (00077) D16$INT2 = $4030 J$MODEM SCROLL INT 2 (J$MODEM)
00004050 (00078) D22$INT1 = $4040 J$ADM INT 1
00004060 (00079) D23$INT1 = $411E J$ADM INT 1

```

J\$START OF MONARRAY FUNCTION DISPATCH TABLE

```

00000700 (00081) POTS = $700
00000710 (00082) J$FLAG SET/CLEAR CONSTANTS

```

```

00000720 (00083) SET = 0*40
00000730 (00084) CLR = 0
00000740 (00085) C0 = 4
00000750 (00086) C1 = 5
00000760 (00087) C2 = 6
00000770 (00088) C3 = 7

```

```

00000780 (00089) J$MODEM-SCROLL INPUT DATA WORD BITS
00000790 (00090) J$LOCAL HANDSET HOOKSWITCH STATE
00000800 (00091) J$DM$SI = $00 J$SIGNAL RATE INDICATOR
00000810 (00092) J$DM$SI = $40 J$INCOMING CALL
00000820 (00093) J$DM$IC = $20 J$CLEAR TO SEND
00000830 (00094) J$DM$CS = $10 J$DATA MODE
00000840 (00095) J$DM$DM = 0 J$RECEIVER READY
00000850 (00096) J$DM$SR = 4 J$SIGNAL QUALITY
00000860 (00097) J$DM$SQ = 2 J$RECEIVE DATA
00000870 (00098) J$DM$SD = 1

```

```

00000880 (00099) J$DM$SI = $502 J$START OF INTEGER SCALAR TABLE IN SNAP EXEC
00000890 (00100) J$DM$SI = $502 J$SYNC-SEARCH LOSE-SYNC THRESHOLD: NUMBER OF
00000900 (00101) J$DM$SI = $502 J$SYNC ERRORS WITHOUT 2 CONSECUTIVE GOOD-SYNC
00000910 (00102) J$DM$SI = $502 J$FRAMES NEEDED TO LOSE SYNC.

```

```

00000920 (00103) J$MODEM-SCROLL OUTPUT DATA WORD BITS
00000930 (00104) J$DM$SI = 0 J$TERMINAL READY
00000940 (00105) J$DM$SI = 0 J$REQUEST TO SEND
00000950 (00106) J$DM$SI = 4 J$SIGNALING RATE
00000960 (00107) J$DM$SI = 2 J$FNO DATA
00000970 (00108) J$DM$SI = 1
00000980 (00109) J$DM$SI = 1
00000990 (00110) J$DM$SI = 1

```

PAGE 5: [000010C0116]000160.MSO.2, 29-Dec-88 15:44:23, Ed: WOLF
SYMBOL DEFINITIONS

```

00000000 (00111) RANDSL = $00000    JLOWER AND UPPER LIMITS FOR ERRPTR
00001C77 (00112) RANDSM = $01C77    J IN MISSIN
00003PCE (00113) SVSPFLGS = $1PCE    JMAP SYSTEM FLAG REGISTER IN PSEUDO-MEMORY
0000000C (00114) TMDITS = 0000000000 JCONTROL BITS USED FOR TRANSMIT MODEN
(00115) J
(00116) JSYNCSOP IS A "REGISTER" IN THE ADAM AND AOM USED FOR CPU-SCROLL
(00117) J COMMUNICATION.
(00118) OPAD0 SYNCSOP,(16 .LS. 16) + (26 .LS. 5)+4
(00119) J
(00120) JTHE SAP (SET ADDRESS FIELD) INSTRUCTION IS ONE OF THE NEW ONES ADDED
(00121) J TO THE CPU BY THE "REV. 10" MICROCODE REVISION.
(00122) OPAD0 SAP,(1 .LS. 14) + (29 .LS. 8) + $EP
(00123)

```

PAGE 6: [000010C0116]000160.MSO.2, 29-Dec-88 15:44:23, Ed: WOLF
PATCHES TO NON-ARRAY AND ARRAY FUNCTION DISPATCH TABLES

```

(00124) "PATCHES TO NON-ARRAY AND ARRAY FUNCTION DISPATCH TABLES
0000000A (00125) HL = FDI$ + (W$ + 121)
00003CCA (00126) ADDR PRORES JFCB 121 (SWAP FUNCTION ADDRESS)
00003CE4 (00127) ADDR PROPAR JFCB 122 (PROPAR)
00003F12 (00128) ADDR CORPAR JFCB 123 (CORPAR)
0000495C (00129) ADDR ADJINT JFCB 124 (ADJINT)
00003000 (00130) ADDR TMDENINT JFCB 125 (TMDINT)
00003067 (00131) ADDR RMDENINT JFCB 126 (RMDINT)
00003C06 (00132) ADDR ADJINT JFCB 127 (ADJINT)
(00133) J
(00134) J
00000150 (00135) HL = AFDT$ORC+3*W$*(189-128) JFCB 188 (DECODE)
0001F3FA (00136) ADDR DCU(R7,1)
0001E3FA (00137) ADDR DCDS(R7,1)
0001021FC (00138) ADDR CSPUSMOS(1,0)
(00139) J
00000156 (00140) HL = AFDT$ORC+3*W$*(189-128) JFCB 189 (DECODE)
0001E3FA (00141) ADDR DCU(R7,1)
0001E3FD4 (00142) ADDR DCDS(R7,1)
0001021FC (00143) ADDR CSPUSMOS(1,0)
(00144) J

```

PAGE 7: (00001<0C416>00016U.WS0.2, 29-Dec-88 15:44:23, Ed: WULF
 PATCHES TO INTERRUPT SERVICE ROUTINES

```

(00145) *PATCHES TO INTERRUPT SERVICE ROUTINES
(00146) ; THESE PATCHES MAKE SCROLL INTERRUPTS GO TO OUR OWN ROUTINES.
(00147) ; NOTE THAT LOADING THIS FILE WILL MAKE THE NORMAL ADAM/ADM/IOS-2 SWAP
(00148) ; FUNCTIONS INOPERATIVE!
00004022 (00149) BL = D16SINT1
00004023 (00150) BL: INCM
00004024 (00151) CALL
00004025 (00152) RET
00004026 (00153) EVEN
00004027 (00154) JMP
00004028 (00155) ;
00004029 (00156) BL = D16SINT2
00004030 (00157) BL: INCM
00004031 (00158) CALL
00004032 (00159) RET
00004033 (00160) EVEN
00004034 (00161) JMP
00004035 (00162) ;
00004036 (00163) BL = D23SINT1
00004037 (00164) BL: INCM
00004038 (00165) CALL
00004039 (00166) RET
00004040 (00167) EVEN
00004041 (00168) JMP
00004042 (00169) ;
00004043 (00170) BL = D23SINT1
00004044 (00171) BL: INCM
00004045 (00172) CALL
00004046 (00173) RET
00004047 (00174) EVEN
00004048 (00175) JMP
00004049 (00176) ;
  
```


PAGE 9: (P000)3<DCAL6>000100.M00.2, 20-Dec-80 15:44:23, Pds WOLF
I/O BUFFER DEFINITIONS

```

00000240 (00230) RSKA = 47944      ;RSINK A BUFFER
00000C20 (00231) RSKB = 48160      ;RSINK B BUFFER
00000C70 (00232) RSKC = 48376      ;D/A "SILENCE" DATA
00000D00 (00233) RADA = 48592      ;D/A A BUFFER
00000E10 (00234) RADB = 48808      ;D/A B BUFFER
00000E20 (00235) ;
00000E30 (00236) ; TWO BUFFERS FOR THE SYNCSEARCH SUBROUTINE ARE DEFINED JUST BELOW THE
00000E40 (00237) ; 16K (FULLWORD) SUBROUTINE ON BUS 1.
00000E50 (00238) RSPF = $7DEC      ;SYNC SEARCH PREVIOUS FRAME (522 HALFWORDS)
00000E60 (00239) RSSS = $7DF6      ;SYNC SEARCH SUM-SYNC COUNTS (SAME)
00000E70 (00240) ;
00000E80 (00241) ; INITIALIZE ALTERNATING SYNC BITS IN THODEM BUFFERS
00000E90 (00242) BL = THDMA
00000EA0 (00243) DATA 0 + THBITS
00000EB0 (00244) BL = THDMB
00000EC0 (00245) DATA 1 + THBITS
00000ED0 (00246)

```



```

(00291) ; MISC. INTEGER MEMORIES (AND INIT VALUES)
(00292) ;
0000053P (00293) RUN = ISVT$+61          ; ENCODER RUN FLAG
0000053P (00294) ; (62 FREE FOR MP1TH USE)
0000053P (00295) RL=RUN
0000053P (00296) DATA 1
0000053P (00297) ;
0000054E (00298) TFCR = ISVT$+76          ; TRANSMITTER FRAME COUNTER (B)
0000054P (00299) RPCR = ISVT$+77          ; RECEIVER FRAME COUNTER (B)
00000550 (00300) RLCTR = ISVT$+78          ; RCVR LOST-SYNC COUNTER (B)
00000551 (00301) RORHX = ISVT$+79          ; LOCAL HANDSET ON-HOOK STATE (B)
00000552 (00302) RSYNC = ISVT$+80          ; STATE OF RMODEM SYNC (B)
00000553 (00303) RROFO = ISVT$+81          ; BEGINNING OF FRAME OFFSET: SYNC BIT
0000054A (00304) ; (62 FREE FOR MP1TH)
0000054A (00305) RL=TAODC
0000054A (00306) DATA 90'B
...
(00307) ;
(00308) ; INTERRUPT COUNTERS
(00309) ;
00000555 (00310) ADCR = ISVT$+83          ; COUNTS ADAM (A/D) INTERRUPTS
00000556 (00311) TMCTR = ISVT$+84          ; COUNTS TMODEM INTERRUPTS
00000557 (00312) RMCTR = ISVT$+85          ; COUNTS RMODEM INTERRUPTS
00000558 (00313) DACR = ISVT$+86          ; COUNTS ADM (D/A) INTERRUPTS
00000555 (00314) RL=ADCTR
00000555 (00315) DATA 40'B
...
(00316) ;
(00317) ; DEBUGGING/DEMONSTRATION AIDS
(00318) ;
00000570 (00319) RMCOR = ISVT$+123          ; SET TO WZ TO TELL CORPAR NOT TO
0000057P (00320) ; (124 FREE FOR MP1TH)          ; CORRECT CHANNEL ERRORS (B)
0000057P (00321) RERRS = ISVT$+125          ; SET TO N>0 TO TELL RMODEM TO
00000570 (00322) ; (126 FREE FOR MP1TH)          ; SIMULATE N CHANNEL ERRORS PER FRAME(B)
00000570 (00323) RL=RMCOR
00000570 (00324) DATA 40'B
...
(00325)

```



```

(00326) *MODFM INTERFACE AND SYSTEM CLOCKS PROGRAMS
(00327) ; JVV, 1A-SEP-79
(00328) ;
(00329) ; THERE ARE TWO ENTRY POINTS:
(00330) ; CLESET: CLOCK RATE SETTING (START ADDRESS = 0)
(00331) ; THIS ENTRY SETS THE DATA AND SAMPLE CLOCKS AND STARTS THEM.
(00332) ; THE 16-BIT CLOCK RATE VALUE IS A LITERAL IN THE INSTRUCTION AT
(00333) ; CLEGO. (TRIED BINDING AN INTEGER TABLE VALUE HERE, BUT BECAUSE
(00334) ; WE USE NO BIDS, LDS$ IN THE SNAP EXEC DOES NO BINDING AT ALL)
(00335) ;
(00336) ; RUNMOD: SET CLOCKS AND RUN MODFM TRANSMIT/RECEIVE (START ADDRESS 1)
(00337) ; THIS ENTRY SETS AND STARTS THE CLOCKS AND THEN RUNS THE MODFM
(00338) ; INTERFACE PROGRAM.
(00339) ;
(00340) ; THESE ENDING ADDRESSES ARE TRUNCATED TO 15 BITS BECAUSE THE LLIT
(00341) ; FIELD OF THE JNE INSTRUCTION IS ONLY THAT BIG.
(00342) ; RMA$END = (RNDMA+MBLNGTH-1) .AND. BITS15
(00343) ; RMB$END = (RNDMB+MBLNGTH-1) .AND. BITS15
(00344) ; RMC$END = (RNDMC+MBLNGTH-1) .AND. BITS15
(00345) ; RMA$END = (RNDMA+MBLNGTH-1) .AND. BITS15
(00346) ; RMB$END = (RNDMB+MBLNGTH-1) .AND. BITS15
(00347) ;
(00348) ; CONSTANT FOR SETTING CLOCK RATES TO MODFM=16000 OPS, SIGNAL
(00349) ; SAMPLE RATE=6621 SAMPLES/SEC.
(00350) ; CLRRATES = $74C6
(00351) ;
(00352) ; REGISTER AND FLAG USAGE:
(00353) ; R0 - RCVR BUFFER SWITCH, SET TO MOD$RA, MOD$RB, OR MOD$RC
(00354) ; R1 - RCVR ADDRESS POINTER
(00355) ; R2 - INTR ADDRESS POINTER
(00356) ; P1 SET SIGNIFIES RCVR DATUM READY
(00357) ; P2 SET SIGNIFIES INTR READY FOR NEXT DATUM
(00358) ; P1 IS USED AS INTR BUFFER SWITCH: CLEAR=>THODEMA, SET=>THODEMB
(00359) ; P2 IS USED TO REMEMBER WHICH START ADDRESS WAS USED.
(00360) ; INT1 IS USED TO SIGNAL END OF TRANSMITTER BUFFER
(00361) ; INT2 IS USED TO SIGNAL END OF RECEIVER BUFFER
(00362) ;
(00363) ; BL = $4000
(00364) ; RA = 0
(00365) ; EVEN
(00366) ; DATA MOD$SZ,0
(00367) ;
(00368) ; MOD$BGM: BEGIN IN$2(RTHODEM)
(00369) ;
(00370) ; CLKSET: SF2,JUMP(CLEGO)
(00371) ; RUNMOD: CF2
(00372) ; CLEGO: LOAD(R0,CLRRATES)
(00373) ; ADDL(R0,0,TP),JIFC(MOD$INIT,P2)
(00374) ; STOP, CF2
(00375) ;
(00376) ; MOD$INIT: LOAD(R0,MOD$RA)
(00377) ; LOAD(R1,RNDMA-1(1),L)
(00378) ;
(00379) ; JENTRY FOR SETTING CLOCKS ONLY
(00380) ; JENTRY FOR CLOCKS AND MODFM
(00381) ; SET DATA AND SAMPLE CLOCKS
(00382) ;
(00383) ; INIT RCVR BUFFER SWITCH
(00384) ; INIT RCVR PTR

```

```

A08 04012 093A0500 (00370)
A09 04014 0A02A969 (00379)
A0A 04016 0C340C9B (00380)
A0B 04018 0B000000 (00381)
A0C 0401C 2E341C00 (00382)
A0D 04020 01300000 (00383)
A0E 04022 11300600 (00384)
A0F 04024 122C001C (00387)
A10 04026 142C0022 (00388)
A11 0402C 165A0001 (00389)
A12 04030 106A2F77 (00390)
A13 04032 1800001C (00391)
A14 04034 1042A747 (00392)
A15 04036 01304000 (00393)
A16 04038 105A0001 (00395)
A17 04040 20000022 (00397)
A18 04042 2147B191 (00398)
A19 04044 013A4000 (00399)
A20 04046 235A0001 (00401)
A21 04048 246A3390 (00402)
A22 0404C 26000015 (00403)
A23 04050 2742A070 (00404)
A24 04052 2A34A000 (00406)
A25 04054 000A0000 (00407)
A26 04056 2C9A0001 (00410)
A27 04058 2E9A2073 (00411)
A28 0405C 3082A073 (00413)
A29 04060 01302100 (00415)
A30 04062 379A0001 (00416)
A31 04064 34A0207D (00417)
A32 04066 00000000 (00418)
A33 04068 3682A969 (00419)
A34 0406C 01302100 (00420)
A35 04070 00000000 (00421)
A36 04072 00000000 (00422)
A37 04074 00000000 (00423)
A38 04076 00000000 (00424)
A39 04078 00000000 (00425)

;SET DIRECTION MAP-TO-MAP
;DISPATCH TO PROPER BUFFER
;FILLER-UPPER
;INCR RPTR AND READ DATUM
;FILLED RMODEMA BUFFER?
;YES, SET SWITCH TO 0
;AND RPTR TO RMODEM
;AND INT LINE 2 TO SAY FULL
;SAME FOR RMODEM

;AND FOR RMODEM

;SET DIRECTION MAP-TO-MODEM
;PI SET MEANS USE TMODEM
;SEND WORD FROM TMODEM
;EMPTIED TMODEM?
;YES, RESET TPTR, FLIP P1,
;INT1 TO SAY BUFFER EMPTY
;SAME FOR TMODEM

;SIZE OF SCROLL PROGRAM IN HW
;PTR-FROM MAP (PTR MEANINGLESS)
;CHNLS=0 : AQ. MODE = DOUBLE. (ALSO)

```

PAGE 14: (R0ND3C0C16)B0N16U.H50.2, 29-Dec-00 15140123, Ed: WULP
 MODEN INTERFACE AND SYSTEM CLOCKS PROGRAMS

04072 0000	(00426)	DATA 0	0103 : 0102
04073 0000	(00427)	DATA 0,0	NO CONTROL REGISTERS
04074 0000			
04075 7777	(00428)	DATA -1,-1,-1,-1,-1	NULL OFFSET, BUFFER CHAIN ANCHORS
04076 7777			
04077 7777			
04078 7777			
04079 7777	(00429)		

```

PAGE 15: (MWD)CCH16>BBI6W.M50.2, 29-04C-00 15:44:23, PDI: WULF
1044 SINGLE-CHANNEL DOUBLE-BUFFERED SAMPLING

(00430) "ADAM SINGLE-CHANNEL DOUBLE-BUFFERED SAMPLING
(00431) J JY, 20 SEPT 79
(00432) J
(00433) J WE SAMPLE ON CHANNEL 1, WHICH IS CONNECTED TO THE G12 SPI.
(00434) J THIS PROGRAM IS FUNCTIONALLY IDENTICAL TO ADMS0, BUT BOUND AT
(00435) J ASSEMBLY TIME.
(00436) J
(00437) AD$END = (TADBA+SLNGTH-1) .AND. BITS15
(00438) AD$END = (TADBB+SLNGTH-1) .AND. BITS15
(00439) J
(00440) ADCNTRL = 2 J INTERNAL TRIGGER, EXTERNAL SAMPLING CLOCK,
(00441) J CH1 CONTROLLED BY SRI, SHORT FLT PT
(00442) J
(00443) J REGISTER AND FLAG USAGE:
(00444) J RB - CONTROLS MUX
(00445) J RI - DATA ADDRESS POINTER
(00446) J FI - THE ADAM USES FI TO TURN SAMPLING ON/OFF
(00447) J INT1 - SIGNALS END OF BUFFER
(00448) J
(00449) RL = $48A0 JBUS 1 ADR FOR ADPROC IOS MODULE
(00450) RA = 0
(00451) EVEN
(00452) DATA AD$SZ,0 JSCROLL PGM SIZE (RW), BIDS 110
(00453) J
(00454) AD$RCH: BEGIN IOS2(ADPROC)
(00455) J
PR JSET DIRECTION ADAM-TO-MAP
LOAD(R0,$0001) JINIT MUX TO CHNL 1
ADD(R0,0,TP)
SPI JSTART SAMPLING
AD$LOOP: LOAD(R1,TADBA-1(1),L) JINIT RPTR TO A/D BUFFER A
ADD(R0,$10,TP) JRELEASE S/H
SUB(R0,$10) JREAD SAMPLE, SEND TO BUFFER
JNE(R1,1,TH) JFILLED BUFFER?
JIFS(AD$STOP,SYNCSOPI),INT1 JYES: INT1 TO SAY FULL, AND CHECK
JIF CSPO SAYS TO STOP
LOAD(R1,TADBB-1(1),L) JINIT RPTR TO A/D BUFFER B
ADD(R0,$10,TP)
SUB(R0,$10)
ADD(R1,1,TH)
JNE(R2,1,AD$END)
JIFC(AD$LOOP,SYNCSOPI), INT1 JSTOP SAMPLING AND HALT
AD$STOP: CFI, STOP
END
(00474) J
(00475) J
(00476) AD$SZ = RL-AD$RCH
(00477) DATA 2
(00478) DATA $0101
(00479) DATA 0
(00480) DATA 3,0
JDIR = TO MAP
JCHNLS=1 : AQ. MODE = DOUBLE
JBI03 : BI02
JLOAD 3 CONTROL RECS STARTING WITH 0

```

PAGE 161 (00007)K0C116>R0M160.N50.2, 29-Dec-88 15:44:23, Ed: WOLF
ADAM SINGLE-CHANNEL DOUBLE-BUFFERED SAMPLING

04002 0000	(00401)	DATA 0	REG. 0: SAMPLE RATE 1 (FOR EXT CLK)
04003 0000	(00402)	DATA 0	REG. 1: SAMPLE RATE 2 NOT USED
04004 0000	(00403)	DATA ADCHNL	REG. 2: ADAM CONTROL BITS
04005 0002	(00404)	DATA -1,-1,-1,-1	BULL OFFSET, BUFFER CHAIN ANCHORS
04006 7777			
04007 7777			
04008 7777			
04009 7777			
0400A 7777	(00405)		

333

PAGE 18: (BBDJ)C0CA16>00M16W.M50.2, 29-Dec-88 15:49:23, Ed: VOL7
 AOM SINGLE-CHANNEL DOUBLE-BUFFERED D/A OUT

04913 0000	(00535)	DATA 0	REG. 0: SAMPLE RATE
04914 0000	(00536)	DATA 0	REG. 1: NOT USED
04915 0002	(00537)	DATA DACCTRL	REG. 2: AOM CONTROL BITS
04916 FFFF	(00538)	DATA -1,-1,-1,-1,-1	NULL OFFSET, BUFFER CHAIN ANCHORS
04917 FFFF			
04918 FFFF			
04919 FFFF			
0491A FFFF			
	(00539)		
	(00540)		

```

(00541) ADAMINT: ADAM INTERRUPT SERVICE ROUTINE
(00542) J JUV, 3-OCT-79
(00543)
(00544) ADAMINT COPIES A/D DATA FROM AN A/D BUFFER TO A TSOURCE BUFFER. IF
(00545) THE LOCAL HANDSET IS ON-HOOK, IT COPIES A LOW-AMPLITUDE "ON-HOOK"
(00546) TONE FROM TADBC INSTEAD, TO CONVEY THIS FACT TO THE REMOTE LISTENER.
(00547) IF THE TSOURCE BUFFER FLAG SHOWS THE BUFFER ISN'T YET AVAILABLE, THE
(00548) FRAME OF A/D DATA IS SIMPLY DISCARDED.
(00549)
0004950 HL = $4950
(00550) JBS 1 ADR FOR ADAMINT CSPO MODULE
(00551) POINTERS TO A/D BUFFERS AND TSOURCE FLAGS AND BUFFER-COPYING ROUTINES.
(00552) THESE BUFFERS AND FLAGS ARE ACCESSED BY MEANS OF POINTER OFFSETS,
(00553) WHICH ARE IN THE INTEGER SCALAR TABLE.
(00554) ADPO IS THE A/D BUFFER POINTER OFFSET (-2,0) INIT TO 0 SO THAT
(00555) IT GETS SET TO TADBA ON THE FIRST CALL
(00556) TSRPO IS THE TSOURCE BUFFER/FLAG POINTER OFFSET (-2,0)
(00557) INIT TO -2 SO THAT IT POINTS TO TSOURCEA ON THE 1ST CALL)
(00558)
EVEN
(00559) ADDR TADBA-2 J A/D BUFFER A (-2 FOR THE BMOVE)
(00560) ADDR TADBA-2 J A/D BUFFER B
(00561) ADPTR: ADDR
(00562)
(00563) ADDR ADSCPA J ROUTINE TO COPY TO TSOURCE A BUFFER
(00564) ADDR ADSCPVB J ROUTINE TO COPY TO TSOURCE B BUFFER
(00565)
(00566) ADDR TSRA J TSOURCE FLAG A
(00567) ADDR TSFB J TSOURCE FLAG B
(00568)
EVEN
(00569) ADAMINT: SETC2,SYSSPLGS JSET C2 TO MARK START OF ADAMINT
(00570) MOVLM R1,ADPO J R1 GETS A/D BUFF PTR OFFSET
(00571) XORL R1,-2 J SWITCH A/D BUFFERS (0,-2,0,...)
(00572) MOVLM R1,ADPO
(00573) CPMZ R1,ADPO
(00574) JNP ADSTN,EQZ JIS THE VOCODER RUNNING?
(00575) MOVLM R3,TSRPO JNO, SO DO NIL
(00576) CPMZ R3,TSRPO J R3 GETS TSOURCE OFFSET
(00577) CPMZ R3,TSRPO JIS THE TSOURCE BUFFER EMPTY?
(00578) JNP AD$DISC,NEZ JNO: DISCARD A/D DATA
(00579) MOVLM R2,ADDBPTR(R1) JYES: GET A/D BUFFER-1 PTR
(00580) CPMZ R2,ADDBPTR(R1) JIS LOCAL HANDSET ON HOOK?
(00581) SKPL EQZ
(00582) MOVLM R2,TADBC-2 JYES: XMIT "ON-HOOK" TONE INSTEAD OF A/D
(00583) EVEN
(00584) CALL R0,TSRBPTR(R3) JCOPY FROM A/D BUFFER TO TSOURCE BUFFER
(00585) MOVLM R1,1
(00586) MOVLM R1,TSRBPTR(R3) JSET TSOURCE FLAG TO SHOW FULL
(00587) XORL R3,-2 J SWITCH TSOURCE BUFFERS (-2,0,-2,...)
(00588) MOVLM R3,TSRPO
(00589) INCH TPCIR JCOUNT FRAME (TO BE) TRANSMITTED
(00590) ADSTN: MOVLM CLR+C2,SYSSPLGS JCLEAR C2 TO MARK END OF ADAMINT
(00591) RETURN
(00592)
(00593) TSOURCE BUFFER ISN'T AVAILABLE, SO WE DISCARD THIS BUFFER OF NEW A/D

```


PAGE 20: (00001<DC16>000160.M50-2, 29-Dec-88 15:44:23, Ed: WOLF
 ADAMINT: ADAM INTERRUPT SERVICE ROUTINE

```

(00594) ; DATA. JUST COUNT THIS FRAME AS DISCARDED.
(00595) ; EVEN
(00596) ; AD$D1SC: INCH TADDC
(00597) ; NOP AD$RTM
(00598) ;
(00599) ; ROUTINES TO COPY FROM THE A/D BUFFER POINTED TO BY (R2)+1
(00600) ; TO A SOURCE BUFFER. A SINGLE MOVE WOULD SUFFICE, BUT THAT WOULD
(00601) ; HOLD OFF INTERRUPTS FOR ABOUT 250 USEC, SO WE BREAK IT UP INTO A FEW,
(00602) ; WHICH IS ALMOST AS FAST, BUT MAKES FOR BETTER INTERRUPT LATENCY.
(00603) ; THE REV-18 CSPD MICROCODE REVISION MODIFIED THE MOVE R,L,MEN SO THAT
(00604) ; (1) R IS LEFT POINTING TO THE LAST SOURCE-WORD MOVED, AND
(00605) ; (2) X IS LEFT WITH ITS ORIGINAL CONTENTS. THIS LETS US USE
(00606) ; THE MOVES AS DONE BELOW WITHOUT RESETTNG R AND X BEFORE EACH ONE.
(00607) ; EVEN
(00608) ; AD$CPY1: MOVIR R4,SL6-1
(00609) ; BMOVEL R2,R4,TSRA
(00610) ; BMOVEL R2,R4,TSRA+2*SL6
(00611) ; BMOVEL R2,R4,TSRA+4*SL6
(00612) ; RETURN
(00613) ;
(00614) ; EVEN
(00615) ; AD$CPY0: MOVIR R4,SL6-1
(00616) ; BMOVEL R2,R4,TSRB
(00617) ; BMOVEL R2,R4,TSRB+2*SL6
(00618) ; BMOVEL R2,R4,TSRB+4*SL6
(00619) ; RETURN
(00620) ;

```

```

(00621) *THODEMINT: THODEM INTERRUPT SERVICE ROUTINE
(00622) ; JYW, 3-OCT-79) MOD FOR DCA16, 11-SEP-80
(00623) ;
(00624) ;THODEMINT COPIES PROTECTED AND BITSTREAMED DATA FROM A TBITS
(00625) ; BUFFER TO A THODEM BUFFER. IF THE TBITS BUFFER FLAG SHOWS THAT
(00626) ; THE BUFFER ISN'T READY YET, A BUFFER OF "FAKE" DATA IS COPIED
(00627) ; FROM TBTC INSTEAD.
(00628) ;
00003000 (00629) BL = $3000 ;PUT THE REST OF THE CSPU CODE IN $3000-$3FFF.
(00630) ;
(00631) ; POINTER OFFSETS:
(00632) ; TBTPD IS FOR TBITS BUFFERS AND FLAGS (-2,0) INIT TO 0 SO IT
(00633) ; LOOKS AT TBTPD THE FIRST TIME)
(00634) ; TBPD IS FOR THODEM BUFFERS (-2,0) INIT TO 0 SO IT GETS SWITCHED
(00635) ; TO THOMA THE FIRST TIME)
(00636) ;
(00637) ;POINTERS TO THE TBITS BUFFERS/FLAGS ARE FOUND IN THE PROTECT.MODULE.
(00638) ;POINTERS TO THODEM-COPYING ROUTINES:
(00639) ;EVEN
00000536 (00640) ;TBTFA ADDR TBTTA ;TBITS BUFFER FLAGS
00000537 (00641) ;TBTTPTR: ADDR TBTPB
(00642) ;
00000538 (00643) ; ADDR THISCPTA ;ROUTINE TO COPY INTO THODEMA
00000539 (00644) ;TMPTR: ADDR THISCPTB ;ROUTINE TO COPY INTO THODEMB
(00645) ;
(00646) ;EVEN
(00647) *THODEMINT:
00000540 (00648) ;MOVLM ;MARK START OF THODEMINT
00000541 (00649) ;MOVLM ;R2 GETS THODEM BUFFER PTR OFFSET
00000542 (00650) ;XORIR R2,-2 ;SWITCH TO NEXT THODEM BUFFER
00000543 (00651) ;MOVLM R2,TBPD
00000544 (00652) ;CMPNZ RUN
00000545 (00653) ;JMP THISTRN,EQZ ;DO NIL IF VACUUM STOPPED
00000546 (00654) ;MOVLM R3,TBTPD ;R3 GETS TBITS PTR OFFSET
00000547 (00655) ;CMPNZ ;IS THE TBITS BUFFER READY (FULL)?
00000548 (00656) ;JMP THISTRN,EQZ ;NO, SO WE RUN FAKE DATA TO THE THODEM
00000549 (00657) ;MOVLM R1,0 ;YES: R1 POINTS TO TBITS BUFFER+1
00000550 (00658) ;SUBIR R1,1 ;MAKE IT POINT TO 1ST WORD
00000551 (00659) ;CALL ;COPY FROM TBITS TO THODEM BUFFER
00000552 (00660) ;MOVLM ;CLEAR TBITS FLAG TO SHOW EMPTY
00000553 (00661) ;XORIR R3,-2 ;SWITCH TBITS BUFFERS (-2,0,-2,...)
00000554 (00662) ;MOVLM R3,TBTPD
00000555 (00663) ;THISTRN: MOVLM CLR+R2,SYS$FLGS ;MARK END OF THODEMINT
00000556 (00664) ;RETURN
(00665) ;
(00666) ;TBITS(X) ISN'T FULL YET, AND WE HAVE TO SEND SOMETHING TO THE
(00667) ; MODEN, SO WE HAVE HANDY A BUFFER (TBTC) OF FAKE TBITS DATA
(00668) ; (CORRESPONDING TO SILENCE), WHICH WE SIMPLY USE INSTEAD.
(00669) ;EVEN
00000570 (00670) ;THIS$FAKE: INCH THFFC ;INCR THODEM FAKE FRAME COUNT
00000571 (00671) ;MOVIR R1,TBTC
00000572 (00672) ;CALL R0,0 ;TMPTR(R2)
00000573 (00673) ;NOP ;RETURN WITHOUT SWITCHING TBITS

```

[illegible]

```

(00705) 'RMODEMINT: RMODEM INTERRUPT SERVICE MODULE
(00706) ; JMW, 25 SEPT 79
(00707) ;
(00708) ; RMODEMINT CHECKS SYNC ON THE BUFFER JUST INPUT OF THE RMODEM SCROLL
(00709) ; PROGRAM, AND IF SYNC IS GOOD, COPIES THE LATEST "FRAME" OF DATA
(00710) ; FROM THE RMODEM BUFFERS TO A RMTS BUFFER. A "FRAME" OF DATA IS
(00711) ; HELD IN WORDS STARTING WITH A SYNC-BIT WORD. IN GENERAL, IT DOES
(00712) ; NOT START AND END ON RMODEM BUFFER BOUNDARIES.
(00713) ; THE STATE OF FRAME SYNC IS GIVEN BY RSYNC.
(00714) ; RSYNC=0 MEANS WE KNOW NOTHING ABOUT SYNC, SO WE MUST CALL SYNCINIT
(00715) ; TO INITIALIZE SYNC-SEARCHING.
(00716) ; RSYNC=-1 MEANS WE ARE STILL IN THE PROCESS OF SCANNING INCOMING
(00717) ; BUFFERS FOR FRAME SYNC (SYNCRCH).
(00718) ; RSYNC=+1 MEANS WE HAVE SYNC, SO WE CALL SYNCUPDATE TO CHECK THE NEW
(00719) ; FRAME FOR CONTINUED SYNC. IF SYNCUPDATE SETS RSYNC TO 0, THEN WE
(00720) ; HAVE LOST SYNC AND WE MUST REINITIALIZE WITH SYNCINIT.
(00721) ; RMODEMINT ALSO:
(00722) ; COPIES THE ON-HOOK BIT OF THE 1ST DATUM IN EACH RMODEM
(00723) ; BUFFER INTO RMTS FOR USE BY THE ADMININT MODULE,
(00724) ; SIMULATES CHANNEL ERRORS IF RERRS IS NONZERO, AND
(00725) ; CHECKS TO BE SURE THAT THE RMTS BUFFER IS AVAILABLE; IF NOT, THEN
(00726) ; THE NEW FRAME OF RMODEM DATA IS SIMPLY DISCARDED.
(00727) ;
(00728) ; POINTER OFFSETS:
(00729) ; RMTPO IS FOR THE RMTS BUFFERS AND FLAG (-2,0) INIT TO -2
(00730) ; SO THAT IT POINTS TO RMTA THE FIRST TIME)
(00731) ; RMPD IS FOR RMODEM BUFFERS (-4,-2,0) INIT TO 0 SO IT GETS SWITCHED
(00732) ; TO RMDNA THE FIRST TIME)
(00733) ;
(00734) ; POINTERS TO RMODEM BUFFERS AND RMTS FLAGS. (POINTERS TO
(00735) ; RMTS BUFFERS ARE IN THE CORPAR MODULE.)
(00736) ; USE RMTPTX(RMTPO) TO POINT TO THE CURRENT RMTS BUFFER AND
(00737) ; RMTPTX(RMTPO) TO POINT TO THE CURRENT FLAG.
(00738) ;
(00739) ;
(00740) ;
(00741) ;
(00742) ;
(00743) ;
(00744) ;
(00745) ;
(00746) ;
(00747) ;
(00748) ;
(00749) ;
(00750) ;
(00751) ;
(00752) ;
(00753) ;
(00754) ;
(00755) ;
(00756) ;
(00757) ;
(00758) ;
(00759) ;
(00760) ;
(00761) ;
(00762) ;
(00763) ;
(00764) ;
(00765) ;
(00766) ;
(00767) ;
(00768) ;
(00769) ;
(00770) ;
(00771) ;
(00772) ;
(00773) ;
(00774) ;
(00775) ;
(00776) ;
(00777) ;
(00778) ;
(00779) ;
(00780) ;
(00781) ;
(00782) ;
(00783) ;
(00784) ;
(00785) ;
(00786) ;
(00787) ;
(00788) ;
(00789) ;
(00790) ;
(00791) ;
(00792) ;
(00793) ;
(00794) ;
(00795) ;
(00796) ;
(00797) ;
(00798) ;
(00799) ;
(00800) ;
(00801) ;
(00802) ;
(00803) ;
(00804) ;
(00805) ;
(00806) ;
(00807) ;
(00808) ;
(00809) ;
(00810) ;
(00811) ;
(00812) ;
(00813) ;
(00814) ;
(00815) ;
(00816) ;
(00817) ;
(00818) ;
(00819) ;
(00820) ;
(00821) ;
(00822) ;
(00823) ;
(00824) ;
(00825) ;
(00826) ;
(00827) ;
(00828) ;
(00829) ;
(00830) ;
(00831) ;
(00832) ;
(00833) ;
(00834) ;
(00835) ;
(00836) ;
(00837) ;
(00838) ;
(00839) ;
(00840) ;
(00841) ;
(00842) ;
(00843) ;
(00844) ;
(00845) ;
(00846) ;
(00847) ;
(00848) ;
(00849) ;
(00850) ;
(00851) ;
(00852) ;
(00853) ;
(00854) ;
(00855) ;
(00856) ;
(00857) ;
(00858) ;
(00859) ;
(00860) ;
(00861) ;
(00862) ;
(00863) ;
(00864) ;
(00865) ;
(00866) ;
(00867) ;
(00868) ;
(00869) ;
(00870) ;
(00871) ;
(00872) ;
(00873) ;
(00874) ;
(00875) ;
(00876) ;
(00877) ;
(00878) ;
(00879) ;
(00880) ;
(00881) ;
(00882) ;
(00883) ;
(00884) ;
(00885) ;
(00886) ;
(00887) ;
(00888) ;
(00889) ;
(00890) ;
(00891) ;
(00892) ;
(00893) ;
(00894) ;
(00895) ;
(00896) ;
(00897) ;
(00898) ;
(00899) ;
(00900) ;
(00901) ;
(00902) ;
(00903) ;
(00904) ;
(00905) ;
(00906) ;
(00907) ;
(00908) ;
(00909) ;
(00910) ;
(00911) ;
(00912) ;
(00913) ;
(00914) ;
(00915) ;
(00916) ;
(00917) ;
(00918) ;
(00919) ;
(00920) ;
(00921) ;
(00922) ;
(00923) ;
(00924) ;
(00925) ;
(00926) ;
(00927) ;
(00928) ;
(00929) ;
(00930) ;
(00931) ;
(00932) ;
(00933) ;
(00934) ;
(00935) ;
(00936) ;
(00937) ;
(00938) ;
(00939) ;
(00940) ;
(00941) ;
(00942) ;
(00943) ;
(00944) ;
(00945) ;
(00946) ;
(00947) ;
(00948) ;
(00949) ;
(00950) ;
(00951) ;
(00952) ;
(00953) ;
(00954) ;
(00955) ;
(00956) ;
(00957) ;
(00958) ;
(00959) ;
(00960) ;
(00961) ;
(00962) ;
(00963) ;
(00964) ;
(00965) ;
(00966) ;
(00967) ;
(00968) ;
(00969) ;
(00970) ;
(00971) ;
(00972) ;
(00973) ;
(00974) ;
(00975) ;
(00976) ;
(00977) ;
(00978) ;
(00979) ;
(00980) ;
(00981) ;
(00982) ;
(00983) ;
(00984) ;
(00985) ;
(00986) ;
(00987) ;
(00988) ;
(00989) ;
(00990) ;
(00991) ;
(00992) ;
(00993) ;
(00994) ;
(00995) ;
(00996) ;
(00997) ;
(00998) ;
(00999) ;
(01000) ;

```

```

03067 1B20 (00758) SKPL LEZ
03068 9021FFFC (00759) MOVIR R2,-4
03069 E200546 (00760) MOVIR R2,RMPO
0306C E20053F (00761) CMPHZ RUN
0306E 001038C2 (00762) JMP RMISRM,EQZ
03070 90943060 (00763) MOVIR R1,0RMPTR(R2)
03072 70220000 (00764) MOVIR R3,R1,LOWSONRE
03074 E0300551 (00765) MOVIR R3,R0RHK
03076 F040057F (00766) MOVIR R4,REARS
03078 1A10 (00767) SKPL EQZ
03079 061038CA (00768) CALL R1,RMISSIM
0307B E200552 (00769) CMPHZ RSYNC
0307D 0000 (00770) EVEN
0307E 001038B6 (00771) JMP RMISIMT,EQZ
03080 003038C0 (00772) JMP RMISRM,LTR
03082 06103C52 (00773) CALL R1,SYNCUPDATE
03084 E200553 (00774) CMPHZ RSYNC
03086 0010380A (00775) JMP RMISLOST,EQZ
03088 F0300547 (00776) MOVIR R3,R0RPO
0308A E2063B50 (00777) CMPHZ R3,R0RTPTR(R2)
0308C 011038C6 (00778) JMP RMISDISC,NEZ
0308E 9094305E (00779) JCOPY FRAME OF BITS (INCLUDING SYNC BIT) FROM PREVIOUS AND CURRENT
03090 FC100553 (00780) J RNDMEN BUFFERS TO R0BITS BUFFER. WE CPT (RNDMEN-R0RPO) BITS
03092 2711 (00781) J FROM (PREV. RNDMEN + R0RPO) TO R0BITS, AND THEN (RNDMEN-R0RPO) BITS FROM
03094 90C63E8C (00782) J (CURRENT RNDMEN) TO (R0BITS + RNDMEN-R0RPO).
03096 EF4030A0 (00783) JNOTE THAT WE JIN ADDRESSES INTO TWO MOVE INSTRUCTIONS BELOW
03098 90500209 (00784) MOVIR R1,0RMPTR-2(R2)
0309A F5500553 (00785) ADDIR R1,R0RPO
0309C 9C400001 (00786) DECR R1,1
0309E EF4030A0 (00787) EVEN
030A0 05100000 (00788) MOVIR R4,0RRT0PTR(R3)
030A2 F0500553 (00789) SAY R4,RMIS0M01
030A4 001038AC (00790) MOVIR R5,RNDMEN-1
030A6 90943060 (00791) SUBIR R5,R0RPO
030A8 2711 (00792) ADDIR R4,1(R5)
030AA 05100000 (00793) SAY R4,RMIS0M02
030AC 25063050 (00794) RMIS0M01: MOVE R1,R5,0
030AE 9431FFFE (00795) MOVIR R5,R0RPO
030B0 00300547 (00796) JMP RMISSTF,EQZ
030B2 0500547 (00797) MOVIR R1,0RMPTR(R2)
030B4 2711 (00798) DECR R1,1
030B6 05100000 (00799) DECR R5,1
030B8 25063050 (00800) RMIS0M02: MOVE R1,R5,0
030BA 9431FFFE (00801) RMISSTF: INCH R3,-2
030BC 00300547 (00802) XORIR R3,R0RPO
030BE 0500547 (00803) MOVIR R3,R0RPO
030B8 25063050 (00804) INCH RPCR
030B4 2000 (00805) HOP RMISRM
030B6 061038EA (00806) EVEN
030B8 2009 (00807) RMISIMT: CALL R1,SYNCINT
030BA 2009 (00808) RMISRM
030BC 2009 (00809) JNOTE THE SYNC-SEARCH
030BE 2009 (00810) J

```

PAGE 25: (RNDJ)DCJ16>RMI160.MSU.2, 29-Dec-88 15:44:22, Ed: WOLF
RMODEMINT: RMODEM INTERRUPT SERVICE MODULE

```

0309 0000 (00011) EVEN
030A 9094305E (00012) RMISLOS: MOVIR R1,ERRPTR-2(R2) ;WE HAVE LOST SYNC!
030B 061030E1 (00013) CALL R1,SYNCRNT ;CALL SYNCRNT ON THE OLDER RMODEM
030C 90943060 (00014) MOVIR R1,ERRPTR(R2) ;BUFFER, THEN SYNCRCH ON THE NEWER ONE
030D 061030C12 (00015) RMISRCH: CALL R1,SYNCRCH
030E 061030C12 (00016) RMISRTH: MOVIR CLR+C2,SYNCSFCS ;MARK END OF RMODEMINT
030F 0000FFCE (00017) RETURN
0310 0670 (00018) ;
0311 (00019) ;THE RMTS BUFFER ISN'T READY (EMPTY) YET, SO WE HAVE NO PLACE TO
0312 (00020) ; PUT THE RMODEM DATA, AND WE HAVE TO DISCARD IT.
0313 0000 (00021) EVEN
0314 0000E50054C (00022) RMISDISC: INCR RMPDC ;COUNT RMODEM FRAME DISCARD
0315 030C 2107 (00023) NOP RMISRTH
0316 (00024) ;
0317 (00025) ;ROUTINE TO SIMULATE CHANNEL ERRORS. WE USE THE EXEC CODE AS A "RANDOM"
0318 (00026) ; NUMBER GENERATOR. WE MAINTAIN A POINTER TO THAT PART OF BUS 1. FOR
0319 (00027) ; EACH ERROR, WE POLL IN 2 WORDS, XOR, AND OFF TO 9 BITS, +10, AND USE
0320 (00028) ; THAT AS AN OFFSET INTO THE CURRENT RMODEM BUFFER, WHERE WE FLIP THE
0321 (00029) ; DATA BIT. THUS WE CAN ERROR ANY OF THE LAST 512 BITS OF THE FRAME.
0322 (00030) ; DOING IT FIVE TIMES PER FRAME GIVES 0.968 ERRORS.
0323 (00031) ;UPON ENTRY, R1 POINTS TO THE START OF THE RMODEM BUFFER, AND R4 HOLDS
0324 (00032) ; THE NUMBER OF ERRORS TO BE MADE.
0325 0000 (00033) EVEN
0326 03CA 2741 (00034) RMISINC: DEC R4,1 ;NUMBER-1 OF ERRORS
0327 0000 (00035) EVEN
0328 005F30E6 (00036) MOVIR R5,ERRPTR ;PICK UP POINTER INTO EXEC CODE
0329 90700001 (00037) MOVIR R7,IDW80 ;WILL BE USED TO FLIP DATA BIT
0330 006A0000 (00038) R1: MOVIR R6,0(R5) ;GET "RANDOM" NUMBER
0331 07002 F46A9001 (00039) INCR R6,1(R5) ;MAKE IT MORE SO
0332 0304 2663 (00040) INCR R6,3 ;AVOID BIAS FOR 0
0333 0305 2653 (00041) INCR R5,3 ;STEP RANDOM NUMBER PTR
0334 0306 9A6001FF (00042) ANDIR R6,$1FF ;MAKE THE RANDOM NUMBER 9 BITS
0335 0308 266A (00043) INCR R6,10 ;GIVE IT THE RANGE 10-521
0336 0309 4C62 (00044) ADDR R6,R1 ;MAKE PTR INTO RMODEM BUFFER
0337 030A F57C0000 (00045) XORRR R7,0(R6) ;FLIP DATA BIT FOR THE ERROR
0338 030C 8C403000 (00046) DJP R4,R1 ;JED BACK IF NOT ENOUGH ERRORS YET
0339 030E 92501C77 (00047) CNPIR R5,RANDSU ;CHECK IF ERRPTR NEEDS RESETING
0340 030E 1A20 (00048) SKPL LE ;RESET ERRPTR TO LOWER VALUE
0341 030F 905000E8 (00049) MOVIR R5,RANDSL
0342 030E 205030E6 (00050) MOVIR R5,ERRPTR
0343 030E 0E70 (00051) RETURN
0344 030E 00E0 (00052) ERRPTR: DATA
0345 (00053)

```

```

(00054) - FRAME SYNCHRONIZATION ROUTINES
(00055) )
(00056) ) MISC. INTERNAL VARIABLES
(00057) ) RESYNC: DATA 0
(00058) ) LASTERR: DATA 0
(00059) ) QLSYNC: DATA 0
(00060) ) HROPD: (1ST 01)
(00061) )
(00062) )
(00063) )
(00064) ) SYNCINIT: INITIALIZE BUFFERS, ETC. FOR SYNC SEARCHING. CALL WITH
(00065) ) R1 POINTING TO RMODEM BUFFER.
(00066) ) ALSO ZEROS THE GAIN WORD OF BOTH RESOURCE BUFFERS AND ZEROS THE
(00067) ) ENTIRE "LAST" RBITS BUFFER, SINCE IF WE JUST LOST SYNC, THOSE
(00068) ) BUFFERS CONTAIN GARBAGE, AND WE DON'T WANT TO RESUME SYNTHESIZING
(00069) ) WITH THEM.
(00070) ) EVEN
(00071) ) SYNCINIT: MOVIR R2, NBLNCTN-1
(00072) ) ADDR R1, R2
(00073) ) SLOOP: MOVIR R3, R1, 1
(00074) ) MOVIR R3, ASSP(R2)
(00075) ) DEC R1
(00076) ) DJP R2, SLOOP
(00077) ) MOVIR R2, ASYNC
(00078) ) RSR+NRSID+WS
(00079) ) RSR+NRSID+WS
(00080) ) MOVIR R3, -2
(00081) ) XOR R3, -2
(00082) ) MOVIR R4, RDTBPTR(R3)
(00083) ) MOVIR R4, 0
(00084) ) INCR R4, WS
(00085) ) SIF R4, SLOOP
(00086) ) DEC R3
(00087) ) MOVIR R2, (NBLNCTN/2)-2
(00088) ) SLOOP: MOVIR R4, R2, 0
(00089) ) MOVIR R4, RSRSS-WS
(00090) ) MOVIR R4, RSRSS+WS
(00091) ) MOVIR R4, R2, 0
(00092) ) RETURN
(00093) )
(00094) )
(00095) ) SYNCRCN: SEARCH INCOMING RMODEM BUFFER FOR SYNC, AND IF SYNC
(00096) ) IS FOUND, SET RSYNC TO +1 AND SET RPOV TO THE FRAME OFFSET
(00097) ) CORRESPONDING TO THE LOGICAL START OF THE FRAME.
(00098) ) CALL WITH R1 POINTING TO THE START OF THE RMODEM BUFFER TO BE
(00099) ) SEARCHED.
(00100) ) REGISTER USAGE:
(00101) ) R1 - PTR TO THE RMODEM BUFFER BEING SEARCHED
(00102) ) R2 - OFFSET INTO THE RMODEM, RSRP, RSRSS BUFFERS
(00103) ) R3 - SCRATCH AC
(00104) ) R4 - MAXCNT, THE MAX. OF RSRSS(I)
(00105) ) R5 - MAX, THE # OF INSTANCES OF MAXCNT IN RSRSS(I)
(00106) ) R6 - IMAX, THE FRAME OFFSET WHERE MAXCNT WAS FOUND

```

(00907) ; VARIABLES:


```

03C4E 20200552 (00960) S5$ND: MOV#M R2,RSYNC      )NO SYNC YET: SET TO -1 (=SEARCH)
03C50 0E70     (00961) RETURN                      ) FOR GOOD MEASURE
              (00962) )
              (00963) )
              (00964) ) SYNCUPDATE: CHECK RMODEM BUFFER TO SEE IF THE DATA BIT AT THE
              (00965) ) EXPECTED SYNC POSITION (START OF BUFFER+RDFO) HAS THE EXPECTED
              (00966) ) VALUE. IF THERE ARE LOSETHR ERRORS WITHOUT 2 CONSECUTIVE CORRECT
              (00967) ) COMPARISONS, CLEAR RSYNC TO SIGNIFY LOSS OF FRAME SYNCHRONIZATION.
              (00968) ) ENTER WITH R1 POINTING TO THE LATEST RMODEM BUFFER. ALSO, THE
              (00969) ) WORD RDFO HOLDS THE OFFSET IN THE FRAME THAT POINTS TO THE
              (00970) ) SYNC WORD TO BE VERIFIED.
              (00971) ) VARIABLES:
              (00972) ) RDFO - BEGINNING OF FRAME OFFSET
              (00973) ) OLDSYNC - SYNC BIT FROM PREVIOUS FRAME
              (00974) ) LASTERR - W2 IF SYNC ERROR ON PREVIOUS FRAME
              (00975) ) ERRSYNC - COUNTS SYNC ERRORS (FROM LOSETHR DOWN TO 0)
              (00976) ) EVEN
              (00977) ) SYNCUPDATE: ADDR#R R1,RDFO
              (00978) ) MOV#R R2,R1
              (00979) ) EVEN
              (00980) ) XOR#R R2,OLDSYNC
              (00981) ) SR#S 0,R2
              (00982) ) NOP S$ERR
              (00983) ) CMP#Z LASTERR
              (00984) ) JNP S$SELF,NEZ
              (00985) ) MOV#R R3,LOSETHR
              (00986) ) MOV#M R3,ERRSYNC
              (00987) ) S$SELF: MOV#M LASTERR
              (00988) ) S$S$VALID: MOV#M R1,OLDSYNC
              (00989) ) HUP
              (00990) )
              (00991) ) EVEN
              (00992) ) S$ERR: MOV#R R2,1
              (00993) ) MOV#M R2,LASTERR
              (00994) ) DEC#M ERRSYNC
              (00995) ) JNP S$LOSE,LEZ
              (00996) ) S$S$VALID: MOV#M R1,LD$NO
              (00997) ) XOR#M R1,OLDSYNC
              (00998) ) RETURN
              (00999) )
              (01000) ) EVEN
              (01001) ) S$S$LOSE: MOV#M RSYNC
              (01002) ) INC#M RLC#R
              (01003) ) RETURN

```

66 L30 E JFF : (50010)
ADMIN: ON INTERRUPT SERVICE ROUTINE. (30010)

PAGE 30: (88ND) <DCAL6> 88M16U.MSO.2, 29-Dec-88 15:44:23, E4: WOLF
 ADJINT: ADM INTERRUPT SERVICE ROUTINE

03C87 0000	(01057) ;		
03C88 904F0023	(01050)	EVEN	
03C89 07200E20	(01059)	DISCPTB: MOVIN	R4,SL6-1
03C8A 07200E20	(01060)	MOVEL	R2,R4,RDAB
03C8C 07200E7F	(01061)	MOVEL	R2,R4,RDAB+2*SL6
03C8E 07200F30	(01062)	MOVEL	R2,R4,RDAB+4*SL6
03C88 0E70	(01063)	RETURN	
	(01064)		

PAGE 31: (PROMD) <DCAL16> 88M16U.MSO.2, 29-DEC-88 15:44:23, ED: VOLF
 PRORES(AB), PROPAR(AB) -- ERROR-PROTECT AND BITSTREAM THE INTR FRAME.
 PRORES(AB), PROPAR(AB) -- ERROR-PROTECT AND BITSTREAM THE INTR FRAME.

```

(01065) )PRORES(AB), PROPAR(AB) -- ERROR-PROTECT AND BITSTREAM THE INTR FRAME.
(01066) )
(01067) )PROTECT TAKES QUANTIZED AND CODED ANALYSIS PARAMETERS AND RESIDUAL
(01068) ) SAMPLES FROM A TSINK BUFFER, AND FORMS A BITSTREAM, INCLUDING
(01069) ) ERROR-PROTECTING CODES FOR SOME BITS OF CERTAIN PARAMETERS, IN
(01070) ) A TSINK BUFFER.
(01071) )THE "PROTECT" MODULE HAS BEEN BROKEN INTO 2 SHORTER ONES, WHICH
(01072) ) CAN BE (MOSTLY) HIDDEN UNDER APU MODULE EXECUTIONS.
(01073) ) PRORES(AB) -- BITSTREAMS THE RESIDUAL SAMPLES (ABOUT 3.7 MS)
(01074) ) PROPAR(AB) -- ERROR-PROTECTS AND BITSTREAMS THE OTHER PARAMETERS
(01075) ) (ABOUT 0.6 MS). PROPAR ALSO ACCUMULATES HISTOGRAM
(01076) ) DATA FOR THE G, DG, P, C1-C3, K1-K6 PARAMETERS.
(01077) )
(01078) )THESE ROUTINES ARE ENTERED WITH R1 POINTING TO THE PC00. 1(R1)
(01079) ) IS THE BUFFER/FLAG/ROUTINE POINTER OFFSET, -2 FOR "A" OR 0 FOR "B".
(01080) )
(01081) )FORMAT OF INPUT (TSINK) BUFFER IS (ONE DATUM PER HALF-WORD):
(01082) ) 216 RESIDUAL SAMPLES (0/2)
(01083) ) GAIN (6/6)
(01084) ) DG1 - DG3 (2/2)
(01085) ) PITCH (7/7)
(01086) ) C1 (2/3)
(01087) ) C2 (3/4)
(01088) ) C3 (2/3)
(01089) ) K1 (5/6)
(01090) ) K2 (4/5)
(01091) ) K3 (3/4)
(01092) ) K4-K6 (2/4)
(01093) )WHERE (M/N) MEANS M HIGH-ORDER BITS PROTECTED OUT OF N TOTAL FOR PARAM.
(01094) )
(01095) )
(01096) )FORMAT OF OUTPUT (BITS) BUFFER IS (ONE BIT PER HALF-WORD):
(01097) ) POSITION FOR THE SYNC BIT (IN THE TWOSEN BUFFER)
(01098) ) RES1 - RES216 (2 BITS EACH, LSO FIRST)
(01099) ) B5432 GAIN (7 BITS, NAMING (7,4) CODE)
(01100) ) B10 GAIN, B10 DG1 (7)
(01101) ) B10 DG2, B10 DG3 (7)
(01102) ) B6543 PITCH (7)
(01103) ) B0 C1 (1)
(01104) ) B2 C1, B210 PITCH (7)
(01105) ) B0 C2 (1)
(01106) ) B321 C2, B1 C1 (7)
(01107) ) B0 C3 (1)
(01108) ) B0 K1 (1)
(01109) ) B54 K1, B21 C3 (7)
(01110) ) B0 K2 (1)
(01111) ) B4321 K2 (7)
(01112) ) B0 K3 (1)
(01113) ) B3 K3, B321 K1 (7)
(01114) ) B32 K4, B21 K3 (7)
(01115) ) B10 K4 (2)
(01116) ) B10 K5 (2)
(01117) ) B10 K6 (2)

```


PAGE 34: (00001<DC116>00M16U-MS0.2, 29-Dec-80 15:44:23, ED: WOLF
PRORES(AB), PROPAR(AB) -- ERROR-PROTECT AND BITSTREAM THE INTR FRAME.

03036 301C	(01224)	POPXI	R1,R6	JC3 (2/3)
03037 0000	(01225)	EVEN		
03038 E50C9C64	(01226)	INCM	CXNIST(R6)	
03039 507C0001	(01227)	MOVR	R7,R6,B0	JEXTRACT B0
0303C 9670000C	(01228)	TORIR	R7,TMBITS	
0303E 342E	(01229)	PUSHXI	R2,R7	JOUTPUT B0 OF C3
0303F 3C61	(01230)	LRS	R6,1	JSHIFT B21 TO B10 POSITION
03040 301A	(01231)			
03041 0000	(01232)	POPXI	R1,R5	JK1 (5/6)
03042 E50A9C6C	(01233)	EVEN		
03043 507A0001	(01234)	INCM	KXNIST(R5)	
03044 507A0001	(01235)	MOVR	R7,R5,B0	JEXTRACT B0 AND OUTPUT IT
03046 9670000C	(01236)	TORIR	R7,TMBITS	
03048 342E	(01237)	PUSHXI	R2,R7	
03049 0000	(01238)	EVEN		
0304A 507A0030	(01239)	MOVR	R7,R5,B54	JEXTRACT B54 OF K1
0304C 3C72	(01240)	LRS	R7,2	JSHIFT IT INTO B32 POSITION
0304D 0000	(01241)	EVEN		
0304E 567C0003	(01242)	TORIR	R7,R6,B10	JADD IN THE SAVED B21 OF C3
03050 8660309C	(01243)	CALL	R6,OUT7	J AND OUTPUT THEN PROTECTED
03052 301C	(01244)			JRS STILL HOLDS B321 OF K1
03053 0000	(01245)	POPXI	R1,R6	JK2 (4/5)
03054 E50C9CAC	(01246)	EVEN		
03055 507C0001	(01247)	INCM	KXNIST(R6)	
03056 507C0001	(01248)	MOVR	R7,R6,B0	JEXTRACT B0 AND OUTPUT IT
03058 9670000C	(01249)	TORIR	R7,TMBITS	
0305A 342E	(01250)	PUSHXI	R2,R7	
0305B 3C61	(01251)	LRS	R6,1	JSHIFT B4321 INTO POSITION
0305C 407C	(01252)	MOVR	R7,R6	J AND OUTPUT PROTECTED
0305D 0000	(01253)	EVEN		
0305E 8660309C	(01254)	CALL	R6,OUT7	
03060 301C	(01255)			
03061 0000	(01256)	POPXI	R1,R6	JK3 (3/4)
03062 E50C9CCC	(01257)	EVEN		
03063 507C0001	(01258)	INCM	KXNIST(R6)	
03064 507C0001	(01259)	MOVR	R7,R6,B0	JEXTRACT L50 AND OUTPUT IT
03066 9670000C	(01260)	TORIR	R7,TMBITS	
03068 342E	(01261)	PUSHXI	R2,R7	
03069 0000	(01262)	EVEN		
0306A 507A000E	(01263)	MOVR	R7,R5,B321	JSET B321 OF K1
0306C 3C71	(01264)	LRS	R7,1	JSHIFT INTO POSITION
0306D 0000	(01265)	EVEN		
0306E 567C0000	(01266)	TORIR	R7,R6,B3	JADD IN THE B3 OF K3
03070 8660309C	(01267)	CALL	R6,OUT7	JOUTPUT B3 K3, B321 K1 PROTECTED
03072 3C61	(01268)	LRS	R6,1	JSHIFT B21 OF K3 INTO B10 POSITION
03073 301A	(01269)			
03074 507A000C	(01270)	POPXI	R1,R5	JFETCH K4 (2/4)
03075 507A000C	(01271)	EVEN		
03076 507A000C	(01272)	INCM	KXNIST(R5)	
03077 567C0003	(01273)	MOVR	R7,R5,B32	JEXTRACT B32 OF K4
03078 567C0003	(01274)	TORIR	R7,R6,B10	JADD THE B21 OF K3
0307A 8660309C	(01275)	CALL	R6,OUT7	JOUTPUT PROTECTED B32 K4, B21 K3
0307C 507A0003	(01276)	MOVR	R7,R5,B10	

```

0307E 86603DA8 (01277) CALL R6,OUT2 JOUTPUT 010 OF K4, UNPROTECTED
03080 301C (01278) POPAT R1,R6 J5 (2/4)
03081 080A (01279) EVEN K5HIST(R5)
03082 558C9C5C (01280) INCH R7,R5,R10
03084 507C0003 (01281) MOVHR R7,R6,R10
03086 86603DA8 (01282) CALL R6,OUT2 JOUTPUT 010 OF K5, UNPROTECTED
03088 301A (01283) POPAT R1,R5 J6 (2/4)
03089 080A (01284) EVEN K5HIST(R5)
0308A 558A9CFC (01285) INCH R7,R5,R10
0308C 507A0003 (01286) MOVHR R7,R6,R10
0308E 86603DA8 (01287) CALL R6,OUT2 JOUTPUT 010 K6, UNPROTECTED
03090 507C000C (01288) LNS R7,2 JSHIFT INTO 010 POSITION
03092 3C72 (01289) EVEN JADD THE 032 OF K6
03093 080A (01290) TONKR R7,R5,R32 JOUTPUT PROTECTED 032 K6, 032 K5
03094 567A000C (01291) CALL R6,OUT2
03096 86603DA8 (01292) MOVLM CLR01,SYSSFLGS
03098 800FFCE (01293) RETURN
0309A 0F70 (01294) JOUT7 TAKES A 4-BIT DATUM IN R7 AND OUTPUTS A 7-BIT HAMMING CODE
0309B 0800 (01295) JNOTE THAT WE MODIFY THE INSTRUCTION AT 0ND07 BY JAMMING THE
0309C 0F203DA4 (01296) J 17-BIT OUTPUT POINTER INTO ITS ADDRESS FIELD.
0309E 4C7E (01297) EVEN
0309F 2627 (01298) OUT7: SAV R2,0ND07 JSTORE OUTPUT PTR IN 0ND07 INSTR.
030A0 90FE30B2 (01299) ADDR R7,R7 JCONVERT R7 TO FULLWORD OFFSET
030A2 90600006 (01300) INCR R2,7 JUPDATE OUTPUT PTR IN R2
030A4 057C0000 (01301) MOVHR R7,0NDPTR(R7) JR7 GETS PTR TO START-1
030A6 0F74 (01302) MOVHR R6,7-1 JWILL TRANSFER 7 WORDS
030A7 0800 (01303) MOVHR R7,R6,0 JMOVE 7 WORDS FROM (R7)+1 TO OUTPUT
030A8 4C7E (01304) RETURN
030A9 2622 (01305) JOUT2 OUTPUTS 2 BITS GIVEN IN R7
030AA 90FF30D2 (01306) EVEN
030AB 0800 (01307) ADDR R7,R7 JCONVERT R7 TO FULLWORD OFFSET
030AC 075F7FE (01308) INCR R2,2 JUPDATE OUTPUT POINTER
030AE 075F7FE (01309) MOVHR R7,0NDPTR(R7) JR7 GETS PTR-1
030B0 0E70 (01310) POPMI R7,-1(R2)
030B1 0800 (01311) RETURN
030B2 08003E01 (01312) JTABLE OF NAMING CODE SEQUENCES. INDEX WITH 2*(4-BIT VALUE)
030B4 08003DD9 (01313) J TO BE PROTECTED; TABLE ENTRY IS PTR-1 TO THE 7-BIT HAMMING
030B6 08003DF3 (01314) J CODEWORD.
030B8 080030F0 (01315) EVEN
030BA 080030E2 (01316) ADDR M000-1
030BC 080030D8 (01317) ADDR M151-1
030BE 080030E1 (01318) ADDR M052-1
030BF 080030E1 (01319) ADDR M103-1
030C0 080030E1 (01320) ADDR M114-1
030C1 080030E1 (01321) ADDR M045-1
030C2 080030E1 (01322) ADDR M146-1
030C3 080030E1 (01323)
030C4 080030E1 (01324)
030C5 080030E1 (01325)
030C6 080030E1 (01326)
030C7 080030E1 (01327)
030C8 080030E1 (01328)
030C9 080030E1 (01329)
030CA 080030E1 (01330)
030CB 080030E1 (01331)
030CC 080030E1 (01332)
030CD 080030E1 (01333)
030CE 080030E1 (01334)
030CF 080030E1 (01335)
030D0 080030E1 (01336)
030D1 080030E1 (01337)
030D2 080030E1 (01338)
030D3 080030E1 (01339)
030D4 080030E1 (01340)
030D5 080030E1 (01341)
030D6 080030E1 (01342)
030D7 080030E1 (01343)
030D8 080030E1 (01344)
030D9 080030E1 (01345)
030DA 080030E1 (01346)
030DB 080030E1 (01347)
030DC 080030E1 (01348)
030DD 080030E1 (01349)
030DE 080030E1 (01350)
030DF 080030E1 (01351)
030E0 080030E1 (01352)
030E1 080030E1 (01353)
030E2 080030E1 (01354)
030E3 080030E1 (01355)
030E4 080030E1 (01356)
030E5 080030E1 (01357)
030E6 080030E1 (01358)
030E7 080030E1 (01359)
030E8 080030E1 (01360)
030E9 080030E1 (01361)
030EA 080030E1 (01362)
030EB 080030E1 (01363)
030EC 080030E1 (01364)
030ED 080030E1 (01365)
030EE 080030E1 (01366)
030EF 080030E1 (01367)
030F0 080030E1 (01368)
030F1 080030E1 (01369)
030F2 080030E1 (01370)
030F3 080030E1 (01371)
030F4 080030E1 (01372)
030F5 080030E1 (01373)
030F6 080030E1 (01374)
030F7 080030E1 (01375)
030F8 080030E1 (01376)
030F9 080030E1 (01377)
030FA 080030E1 (01378)
030FB 080030E1 (01379)
030FC 080030E1 (01380)
030FD 080030E1 (01381)
030FE 080030E1 (01382)
030FF 080030E1 (01383)
03100 080030E1 (01384)
03101 080030E1 (01385)
03102 080030E1 (01386)
03103 080030E1 (01387)
03104 080030E1 (01388)
03105 080030E1 (01389)
03106 080030E1 (01390)
03107 080030E1 (01391)
03108 080030E1 (01392)
03109 080030E1 (01393)
0310A 080030E1 (01394)
0310B 080030E1 (01395)
0310C 080030E1 (01396)
0310D 080030E1 (01397)
0310E 080030E1 (01398)
0310F 080030E1 (01399)
03110 080030E1 (01400)
03111 080030E1 (01401)
03112 080030E1 (01402)
03113 080030E1 (01403)
03114 080030E1 (01404)
03115 080030E1 (01405)
03116 080030E1 (01406)
03117 080030E1 (01407)
03118 080030E1 (01408)
03119 080030E1 (01409)
0311A 080030E1 (01410)
0311B 080030E1 (01411)
0311C 080030E1 (01412)
0311D 080030E1 (01413)
0311E 080030E1 (01414)
0311F 080030E1 (01415)
03120 080030E1 (01416)
03121 080030E1 (01417)
03122 080030E1 (01418)
03123 080030E1 (01419)
03124 080030E1 (01420)
03125 080030E1 (01421)
03126 080030E1 (01422)
03127 080030E1 (01423)
03128 080030E1 (01424)
03129 080030E1 (01425)
0312A 080030E1 (01426)
0312B 080030E1 (01427)
0312C 080030E1 (01428)
0312D 080030E1 (01429)
0312E 080030E1 (01430)
0312F 080030E1 (01431)
03130 080030E1 (01432)
03131 080030E1 (01433)
03132 080030E1 (01434)
03133 080030E1 (01435)
03134 080030E1 (01436)
03135 080030E1 (01437)
03136 080030E1 (01438)
03137 080030E1 (01439)
03138 080030E1 (01440)
03139 080030E1 (01441)
0313A 080030E1 (01442)
0313B 080030E1 (01443)
0313C 080030E1 (01444)
0313D 080030E1 (01445)
0313E 080030E1 (01446)
0313F 080030E1 (01447)
03140 080030E1 (01448)
03141 080030E1 (01449)
03142 080030E1 (01450)
03143 080030E1 (01451)
03144 080030E1 (01452)
03145 080030E1 (01453)
03146 080030E1 (01454)
03147 080030E1 (01455)
03148 080030E1 (01456)
03149 080030E1 (01457)
0314A 080030E1 (01458)
0314B 080030E1 (01459)
0314C 080030E1 (01460)
0314D 080030E1 (01461)
0314E 080030E1 (01462)
0314F 080030E1 (01463)
03150 080030E1 (01464)
03151 080030E1 (01465)
03152 080030E1 (01466)
03153 080030E1 (01467)
03154 080030E1 (01468)
03155 080030E1 (01469)
03156 080030E1 (01470)
03157 080030E1 (01471)
03158 080030E1 (01472)
03159 080030E1 (01473)
0315A 080030E1 (01474)
0315B 080030E1 (01475)
0315C 080030E1 (01476)
0315D 080030E1 (01477)
0315E 080030E1 (01478)
0315F 080030E1 (01479)
03160 080030E1 (01480)
03161 080030E1 (01481)
03162 080030E1 (01482)
03163 080030E1 (01483)
03164 080030E1 (01484)
03165 080030E1 (01485)
03166 080030E1 (01486)
03167 080030E1 (01487)
03168 080030E1 (01488)
03169 080030E1 (01489)
0316A 080030E1 (01490)
0316B 080030E1 (01491)
0316C 080030E1 (01492)
0316D 080030E1 (01493)
0316E 080030E1 (01494)
0316F 080030E1 (01495)
03170 080030E1 (01496)
03171 080030E1 (01497)
03172 080030E1 (01498)
03173 080030E1 (01499)
03174 080030E1 (01500)
03175 080030E1 (01501)
03176 080030E1 (01502)
03177 080030E1 (01503)
03178 080030E1 (01504)
03179 080030E1 (01505)
0317A 080030E1 (01506)
0317B 080030E1 (01507)
0317C 080030E1 (01508)
0317D 080030E1 (01509)
0317E 080030E1 (01510)
0317F 080030E1 (01511)
03180 080030E1 (01512)
03181 080030E1 (01513)
03182 080030E1 (01514)
03183 080030E1 (01515)
03184 080030E1 (01516)
03185 080030E1 (01517)
03186 080030E1 (01518)
03187 080030E1 (01519)
03188 080030E1 (01520)
03189 080030E1 (01521)
0318A 080030E1 (01522)
0318B 080030E1 (01523)
0318C 080030E1 (01524)
0318D 080030E1 (01525)
0318E 080030E1 (01526)
0318F 080030E1 (01527)
03190 080030E1 (01528)
03191 080030E1 (01529)
03192 080030E1 (01530)
03193 080030E1 (01531)
03194 080030E1 (01532)
03195 080030E1 (01533)
03196 080030E1 (01534)
03197 080030E1 (01535)
03198 080030E1 (01536)
03199 080030E1 (01537)
0319A 080030E1 (01538)
0319B 080030E1 (01539)
0319C 080030E1 (01540)
0319D 080030E1 (01541)
0319E 080030E1 (01542)
0319F 080030E1 (01543)
031A0 080030E1 (01544)
031A1 080030E1 (01545)
031A2 080030E1 (01546)
031A3 080030E1 (01547)
031A4 080030E1 (01548)
031A5 080030E1 (01549)
031A6 080030E1 (01550)
031A7 080030E1 (01551)
031A8 080030E1 (01552)
031A9 080030E1 (01553)
031AA 080030E1 (01554)
031AB 080030E1 (01555)
031AC 080030E1 (01556)
031AD 080030E1 (01557)
031AE 080030E1 (01558)
031AF 080030E1 (01559)
031B0 080030E1 (01560)
031B1 080030E1 (01561)
031B2 080030E1 (01562)
031B3 080030E1 (01563)
031B4 080030E1 (01564)
031B5 080030E1 (01565)
031B6 080030E1 (01566)
031B7 080030E1 (01567)
031B8 080030E1 (01568)
031B9 080030E1 (01569)
031BA 080030E1 (01570)
031BB 080030E1 (01571)
031BC 080030E1 (01572)
031BD 080030E1 (01573)
031BE 080030E1 (01574)
031BF 080030E1 (01575)
031C0 080030E1 (01576)
031C1 080030E1 (01577)
031C2 080030E1 (01578)
031C3 080030E1 (01579)
031C4 080030E1 (01580)
031C5 080030E1 (01581)
031C6 080030E1 (01582)
031C7 080030E1 (01583)
031C8 080030E1 (01584)
031C9 080030E1 (01585)
031CA 080030E1 (01586)
031CB 080030E1 (01587)
031CC 080030E1 (01588)
031CD 080030E1 (01589)
031CE 080030E1 (01590)
031CF 080030E1 (01591)
031D0 080030E1 (01592)
031D1 080030E1 (01593)
031D2 080030E1 (01594)
031D3 080030E1 (01595)
031D4 080030E1 (01596)
031D5 080030E1 (01597)
031D6 080030E1 (01598)
031D7 080030E1 (01599)
031D8 080030E1 (01600)
031D9 080030E1 (01601)
031DA 080030E1 (01602)
031DB 080030E1 (01603)
031DC 080030E1 (01604)
031DD 080030E1 (01605)
031DE 080030E1 (01606)
031DF 080030E1 (01607)
031E0 080030E1 (01608)
031E1 080030E1 (01609)
031E2 080030E1 (01610)
031E3 080030E1 (01611)
031E4 080030E1 (01612)
031E5 080030E1 (01613)
031E6 080030E1 (01614)
031E7 080030E1 (01615)
031E8 080030E1 (01616)
031E9 080030E1 (01617)
031EA 080030E1 (01618)
031EB 080030E1 (01619)
031EC 080030E1 (01620)
031ED 080030E1 (01621)
031EE 080030E1 (01622)
031EF 080030E1 (01623)
031F0 080030E1 (01624)
031F1 080030E1 (01625)
031F2 080030E1 (01626)
031F3 080030E1 (01627)
031F4 080030E1 (01628)
031F5 080030E1 (01629)
031F6 080030E1 (01630)
031F7 080030E1 (01631)
031F8 080030E1 (01632)
031F9 080030E1 (01633)
031FA 080030E1 (01634)
031FB 080030E1 (01635)
031FC 080030E1 (01636)
031FD 080030E1 (01637)
031FE 080030E1 (01638)
031FF 080030E1 (01639)
03200 080030E1 (01640)
03201 080030E1 (01641)
03202 080030E1 (01642)
03203 080030E1 (01643)
03204 080030E1 (01644)
03205 080030E1 (01645)
03206 080030E1 (01646)
03207 080030E1 (01647)
03208 080030E1 (01648)
03209 080030E1 (01649)
0320A 080030E1 (01650)
0320B 080030E1 (01651)
0320C 080030E1 (01652)
0320D 080030E1 (01653)
0320E 080030E1 (01654)
0320F 080030E1 (01655)
03210 080030E1 (01656)
03211 080030E1 (01657)
03212 080030E1 (01658)
03213 080030E1 (01659)
03214 080030E1 (01660)
03215 080030E1 (01661)
03216 080030E1 (01662)
03217 080030E1 (01663)
03218 080030E1 (01664)
03219 080030E1 (01665)
0321A 080030E1 (01666)
0321B 080030E1 (01667)
0321C 080030E1 (01668)
0321D 080030E1 (01669)
0321E 080030E1 (01670)
0321F 080030E1 (01671)
03220 080030E1 (01672)
03221 080030E1 (01673)
03222 080030E1 (01674)
03223 080030E1 (01675)
03224 080030E1 (01676)
03225 080030E1 (01677)
03226 080030E1 (01678)
03227 080030E1 (01679)
03228 080030E1 (01680)
03229 080030E1 (01681)
0322A 080030E1 (01682)
0322B 080030E1 (01683)
0322C 080030E1 (01684)
0322D 080030E1 (01685)
0322E 080030E1 (01686)
0322F 080030E1 (01687)
03230 080030E1 (01688)
03231 080030E1 (01689)
03232 080030E1 (01690)
03233 080030E1 (01691)
03234 080030E1 (01692)
03235 080030E1 (01693)
03236 080030E1 (01694)
03237 080030E1 (01695)
03238 080030E1 (01696)
03239 080030E1 (01697)
0323A 080030E1 (01698)
0323B 080030E1 (01699)
0323C 080030E1 (01700)
0323D 080030E1 (01701)
0323E 080030E1 (01702)
0323F 080030E1 (01703)
03240 080030E1 (01704)
03241 080030E1 (01705)
03242 080030E1 (01706)
03243 080030E1 (01707)
03244 080030E1 (01708)
03245 080030E1 (01709)
03246 080030E1 (01710)
03247 080030E1 (01711)
03248 080030E1 (01712)
03249 080030E1 (01713)
0324A 080030E1 (01714)
0324B 080030E1 (01715)
0324C 080030E1 (01716)
0324D 080030E1 (01717)
0324E 080030E1 (01718)
0324F 080030E1 (01719)
03250 080030E1 (01720)
03251 080030E1 (01721)
03252 080030E1 (01722)
03253 080030E1 (01723)
03254 080030E1 (01724)
03255 080030E1 (01725)
03256 080030E1 (01726)
03257 080030E1 (01727)
03258 080030E1 (01728)
03259 080030E1 (01729)
0325A 080030E1 (01730)
0325B 080030E1 (01731)
0325C 080030E1 (01732)
0325D 080030E1 (01733)
0325E 080030E1 (01734)
0325F 080030E1 (01735)
03260 080030E1 (01736)
03261 080030E1 (01737)
03262 080030E1 (01738)
03263 080030E1 (01739)
03264 080030E1 (01740)
03265 080030E1 (01741)
03266 080030E1 (01742)
03267 080030E1 (01743)
03268 080030E1 (01744)
03269 080030E1 (01745)
0326A 080030E1 (01746)
0326B 080030E1 (01747)
0326C 080030E1 (01748)
0326D 080030E1 (01749)
0326E 080030E1 (01750)
0326F 080030E1 (01751)
03270 080030E1 (01752)
03271 080030E1 (01753)
03272 080030E1 (01754)
03273 080030E1 (01755)
03274 080030E1 (01756)
03275 080030E1 (01757)
03276 080030E1 (01758)
03277 080030E1 (01759)
03278 080030E1 (01760)
03279 080030E1 (01761)
0327A 080030E1 (01762)
0327B 080030E1 (01763)
0327C 080030E1 (01764)
0327D 080030E1 (01765)
0327E 080030E1 (01766)
0327F 080030E1 (01767)
03280 080030E1 (01768)
03281 080030E1 (01769)
03282 080030E1 (01770)
03283 080030E1 (01771)
03284 080030E1 (01772)
03285 080030E1 (01773)
03286 080030E1 (01774)
03287 080030E1 (01775)
03288 080030E1 (01776)
03289 080030E1 (01777)
0328A 080030E1 (01778)
0328B 080030E1 (01779)
0328C 080030E1 (01780)
0328D 080030E1 (01781)
0328E 080030E1 (01782)
0328F 080030E1 (01783)
03290 080030E1 (01784)
03291 080030E1 (01785)
03292 080030E1 (01786)
03293 080030E1 (01787)
03294 080030E1 (01788)
03295 080030E1 (01789)
03296 080030E1 (01790)
03297 080030E1 (01791)
03298 080030E1 (01792)
03299 080030E1 (01793)
0329A 080030E1 (01794)
0329B 080030E1 (01795)
0329C 080030E1 (01796)
0329D 080030E1 (01797)
0329E 080030E1 (01798)
0329F 080030E1 (01799)
032A0 080030E1 (01800)
032A1 080030E1 (01801)
032A2 080030E1 (01802)
032A3 080030E1 (01803)
032A4 080030E1 (01804)
032A5 080030E1 (01805)
032A6 080030E1 (01806)
032A7 080030E1 (01807)
032A8 080030E1 (01808)
032A9 080030E1 (01809)
032AA 080030E1 (01810)
032AB 080030E1 (01811)
032AC 080030E1 (01812)
032AD 080030E1 (01813)
032AE 080030E1 (01814)
032AF 080030E1 (01815)
032B0 080030
```


PAGE 36: (PMB)<DC116>MM16V.MS0.2, 29-Dec-88 15:44:23, Ed: WMLF
 PROPS(AB), PROPAN(AB) -- ERROR-PROTECT AND BITSTREAM THE INTR FRAME.

03DC0 00030DFA (01370) ADDR M017-1
 03DC2 00030DFE (01331) ADDR M160-1
 03DC4 00030DE3 (01332) ADDR M031-1
 03DC6 00030DE7 (01333) ADDR M132-1
 03DC8 00030DE4 (01334) ADDR M063-1
 03DCA 00030DFC (01335) ADDR M074-1
 03DCC 00030DF2 (01336) ADDR M125-1
 03DCE 00030D00 (01337) ADDR M026-1
 03DD0 00030DE9 (01338) ADDR M177-1
 (01339)

TABLE OF PTR-1 TO 2-BIT SEQUENCES

03DD2 00030F03 (01340) PTR2: ADDR Q0-1
 03DD4 00030E08 (01341) ADDR Q1-1
 03DD6 00030D07 (01342) ADDR Q2-1
 03DD8 00030DF1 (01343) ADDR Q3-1
 (01344)
 (01345)

TABLE OF CODE SEQUENCES.

03DDA 0000 (01346) H151: DATA 1 + TM01TS
 03DDB 0000 (01347) H151: DATA 1 + TM01TS
 03DDC 000C (01348) P51: DATA 1 + TM01TS
 03DD0 000C (01349) H045: DATA 0 + TM01TS
 03DE0 000C (01350) H045: DATA 1 + TM01TS
 03DE2 000C (01351) H026: DATA 0 + TM01TS
 03DE4 000C (01352) P2: DATA 0 + TM01TS
 03DE6 000C (01353) Q2: DATA 1 + TM01TS
 03DE8 000C (01354) DATA 0 + TM01TS
 03DE1 000C (01355) H146: DATA 1 + TM01TS
 03DE2 0000 (01356) H144: DATA 1 + TM01TS
 03DE3 0000 (01357) H031: DATA 0 + TM01TS
 03DE4 000C (01358) H063: DATA 0 + TM01TS
 03DE5 000C (01359) P6: DATA 1 + TM01TS
 03DE6 0000 (01360) DATA 1 + TM01TS
 03DE7 0000 (01361) P1: DATA 0 + TM01TS
 03DE8 000C (01362) Q1: DATA 0 + TM01TS
 03DE9 000C (01363) H177: DATA 1 + TM01TS
 03DEA 0000 (01364) P7: DATA 1 + TM01TS
 03DEB 0000 (01365) DATA 1 + TM01TS
 03DEC 0000 (01366) DATA 1 + TM01TS
 03DED 0000 (01367) DATA 1 + TM01TS
 03DEF 0000 (01368) DATA 1 + TM01TS
 03DF0 0000 (01369) H132: DATA 1 + TM01TS
 03DF1 000C (01370) P3: DATA 0 + TM01TS
 03DF2 0000 (01371) Q3: DATA 1 + TM01TS
 03DF3 0000 (01372) H125: DATA 1 + TM01TS
 03DF4 000C (01373) H052: DATA 0 + TM01TS
 03DF5 0000 (01374) DATA 1 + TM01TS
 03DF6 000C (01375) DATA 0 + TM01TS
 03DF7 0000 (01376) DATA 1 + TM01TS
 03DF8 000C (01377) DATA 0 + TM01TS
 03DF9 0000 (01378) H103: DATA 1 + TM01TS
 03DFA 000C (01379) DATA 0 + TM01TS
 03DFB 000C (01380) H017: DATA 0 + TM01TS
 03DFC 000C (01381) DATA 0 + TM01TS
 03DFD 000C (01382) H074: DATA 0 + TM01TS

I

— — —

PAGE 38: (0000)<DC016>000160,MS0.2, 29-Dec-88 15:44:23, Ed: WOLF
 COMPAR(AB): UNBITSTREAM, ERROR-CORRECT, AND DECODE PARAMETERS

```

(01395) "COMPAB(AB): UNBITSTREAM, ERROR-CORRECT, AND DECODE PARAMETERS
(01396) )
(01397) )CORPAR TAKES A FRAME OF BITSTREAM (R0ITS BUFFER) FROM THE
(01398) ) RECEIVER MODERN AND TURNS IT INTO PARAMETERS,
(01399) ) DOING ERROR-CORRECTING DECODING FOR THE PROTECTED DATA BITS. EACH
(01400) ) UNBITSTREAMED AND CORRECTED VALUE IS THEN LOOKED UP IN THE
(01401) ) APPROPRIATE DECODING TABLE, SO THAT THE OUTPUT (R0SOURCE) BUFFER IS
(01402) ) FULL-WORD FLOATING POINT VALUES READY FOR THE SYNTHESIS PROCESS.
(01403) ) CORPAR DOES NOT TOUCH THE RESIDUAL SAMPLES) THEY GET UNBITSTREAMED
(01404) ) AND DECODED BY THE AP MODULES DECODA/DECODEB.
(01405) )CORPAR NO LONGER RUNS AT INTERRUPT LEVEL, BUT IS INVOKED VIA PCB.
(01406) )
(01407) )THIS ROUTINE IS ENTERED (VIA PCB) WITH R1 POINTING TO THE
(01408) ) PCB. 1(R1) IS THE BUFFER/FLAG POINTER OFFSET, EQUAL TO -2 FOR "A"
(01409) ) OR 0 FOR "B".
(01410) )REGISTER USAGE:
(01411) ) R1 = PTR-1 TO THE INPUT (R0ITS) BUFFER (USED BY POPP'S)
(01412) ) R2 = PTR TO THE OUTPUT (R0SOURCE) BUFFER (USED IN PUSHMIL'S)
(01413) ) R3-R5 = SCRATCH IN PARAMETER DECODING
(01414) ) R6 = SCRATCH IN PARAM DECODING AND IN GET74, ETC.
(01415) ) R7 = GET74, ETC. RETURN VALUES IN HERE
(01416) )
(01417) )THE FORMAT OF THE INPUT (R0ITS) BUFFER IS IDENTICAL TO THE TRITS
(01418) ) BUFFER, WHICH IS DESCRIBED ABOVE IN THE PHORES/PROPAR DOCUMENTATION.
(01419) ) THE FORMAT OF THE OUTPUT (R0SOURCE) BUFFER IS LIKE THAT OF THE
(01420) ) TSIME BUFFER (ALSO DESCRIBED ABOVE), EXCEPT THAT EACH DATUM OCCUPIES
(01421) ) A FULL-WORD INSTEAD OF A HALF-WORD, BECAUSE THESE ARE DECODED
(01422) ) FLOATING POINT VALUES.
(01423) )
(01424) )POINTERS TO BUFFERS. THE R0ITS POINTERS ARE ALSO USED BY RMODEMINT.
(01425) )
(01426) ) R0TA R0TB R0ITS BUFFERS
(01427) ) R0TAPTR: ADDR R0TB
(01428) )
(01429) ) ADDR R0RA+RRESID*W$ R0SOURCE BUFFER PTRS TO PARAMETERS
(01430) ) R0RBPTR: ADDR R0RB+RRESID*W$ )(STEPPING OVER THE RESIDUAL)
(01431) )
(01432) )
(01433) )
(01434) )CORPAR: MOVLM
(01435) ) W0VNR
(01436) ) MOVIR
(01437) )
(01438) ) ADDR
(01439) ) MOVIR
(01440) ) CALL
(01441) ) LLS
(01442) ) MOVRR
(01443) )
(01444) ) CALL
(01445) ) MOVRR
(01446) ) LRS
(01447) ) TORR
(01448) )
(01449) )
(01450) )
(01451) )
(01452) )
(01453) )
(01454) )
(01455) )
(01456) )
(01457) )
(01458) )
(01459) )
(01460) )
(01461) )
(01462) )
(01463) )
(01464) )
(01465) )
(01466) )
(01467) )
(01468) )
(01469) )
(01470) )
(01471) )
(01472) )
(01473) )
(01474) )
(01475) )
(01476) )
(01477) )
(01478) )
(01479) )
(01480) )
(01481) )
(01482) )
(01483) )
(01484) )
(01485) )
(01486) )
(01487) )
(01488) )
(01489) )
(01490) )
(01491) )
(01492) )
(01493) )
(01494) )
(01495) )
(01496) )
(01497) )
(01498) )
(01499) )
(01500) )
(01501) )
(01502) )
(01503) )
(01504) )
(01505) )
(01506) )
(01507) )
(01508) )
(01509) )
(01510) )
(01511) )
(01512) )
(01513) )
(01514) )
(01515) )
(01516) )
(01517) )
(01518) )
(01519) )
(01520) )
(01521) )
(01522) )
(01523) )
(01524) )
(01525) )
(01526) )
(01527) )
(01528) )
(01529) )
(01530) )
(01531) )
(01532) )
(01533) )
(01534) )
(01535) )
(01536) )
(01537) )
(01538) )
(01539) )
(01540) )
(01541) )
(01542) )
(01543) )
(01544) )
(01545) )
(01546) )
(01547) )
(01548) )
(01549) )
(01550) )
(01551) )
(01552) )
(01553) )
(01554) )
(01555) )
(01556) )
(01557) )
(01558) )
(01559) )
(01560) )
(01561) )
(01562) )
(01563) )
(01564) )
(01565) )
(01566) )
(01567) )
(01568) )
(01569) )
(01570) )
(01571) )
(01572) )
(01573) )
(01574) )
(01575) )
(01576) )
(01577) )
(01578) )
(01579) )
(01580) )
(01581) )
(01582) )
(01583) )
(01584) )
(01585) )
(01586) )
(01587) )
(01588) )
(01589) )
(01590) )
(01591) )
(01592) )
(01593) )
(01594) )
(01595) )
(01596) )
(01597) )
(01598) )
(01599) )
(01600) )
(01601) )
(01602) )
(01603) )
(01604) )
(01605) )
(01606) )
(01607) )
(01608) )
(01609) )
(01610) )
(01611) )
(01612) )
(01613) )
(01614) )
(01615) )
(01616) )
(01617) )
(01618) )
(01619) )
(01620) )
(01621) )
(01622) )
(01623) )
(01624) )
(01625) )
(01626) )
(01627) )
(01628) )
(01629) )
(01630) )
(01631) )
(01632) )
(01633) )
(01634) )
(01635) )
(01636) )
(01637) )
(01638) )
(01639) )
(01640) )
(01641) )
(01642) )
(01643) )
(01644) )
(01645) )
(01646) )
(01647) )
(01648) )
(01649) )
(01650) )
(01651) )
(01652) )
(01653) )
(01654) )
(01655) )
(01656) )
(01657) )
(01658) )
(01659) )
(01660) )
(01661) )
(01662) )
(01663) )
(01664) )
(01665) )
(01666) )
(01667) )
(01668) )
(01669) )
(01670) )
(01671) )
(01672) )
(01673) )
(01674) )
(01675) )
(01676) )
(01677) )
(01678) )
(01679) )
(01680) )
(01681) )
(01682) )
(01683) )
(01684) )
(01685) )
(01686) )
(01687) )
(01688) )
(01689) )
(01690) )
(01691) )
(01692) )
(01693) )
(01694) )
(01695) )
(01696) )
(01697) )
(01698) )
(01699) )
(01700) )
(01701) )
(01702) )
(01703) )
(01704) )
(01705) )
(01706) )
(01707) )
(01708) )
(01709) )
(01710) )
(01711) )
(01712) )
(01713) )
(01714) )
(01715) )
(01716) )
(01717) )
(01718) )
(01719) )
(01720) )
(01721) )
(01722) )
(01723) )
(01724) )
(01725) )
(01726) )
(01727) )
(01728) )
(01729) )
(01730) )
(01731) )
(01732) )
(01733) )
(01734) )
(01735) )
(01736) )
(01737) )
(01738) )
(01739) )
(01740) )
(01741) )
(01742) )
(01743) )
(01744) )
(01745) )
(01746) )
(01747) )
(01748) )
(01749) )
(01750) )
(01751) )
(01752) )
(01753) )
(01754) )
(01755) )
(01756) )
(01757) )
(01758) )
(01759) )
(01760) )
(01761) )
(01762) )
(01763) )
(01764) )
(01765) )
(01766) )
(01767) )
(01768) )
(01769) )
(01770) )
(01771) )
(01772) )
(01773) )
(01774) )
(01775) )
(01776) )
(01777) )
(01778) )
(01779) )
(01780) )
(01781) )
(01782) )
(01783) )
(01784) )
(01785) )
(01786) )
(01787) )
(01788) )
(01789) )
(01790) )
(01791) )
(01792) )
(01793) )
(01794) )
(01795) )
(01796) )
(01797) )
(01798) )
(01799) )
(01800) )
(01801) )
(01802) )
(01803) )
(01804) )
(01805) )
(01806) )
(01807) )
(01808) )
(01809) )
(01810) )
(01811) )
(01812) )
(01813) )
(01814) )
(01815) )
(01816) )
(01817) )
(01818) )
(01819) )
(01820) )
(01821) )
(01822) )
(01823) )
(01824) )
(01825) )
(01826) )
(01827) )
(01828) )
(01829) )
(01830) )
(01831) )
(01832) )
(01833) )
(01834) )
(01835) )
(01836) )
(01837) )
(01838) )
(01839) )
(01840) )
(01841) )
(01842) )
(01843) )
(01844) )
(01845) )
(01846) )
(01847) )
(01848) )
(01849) )
(01850) )
(01851) )
(01852) )
(01853) )
(01854) )
(01855) )
(01856) )
(01857) )
(01858) )
(01859) )
(01860) )
(01861) )
(01862) )
(01863) )
(01864) )
(01865) )
(01866) )
(01867) )
(01868) )
(01869) )
(01870) )
(01871) )
(01872) )
(01873) )
(01874) )
(01875) )
(01876) )
(01877) )
(01878) )
(01879) )
(01880) )
(01881) )
(01882) )
(01883) )
(01884) )
(01885) )
(01886) )
(01887) )
(01888) )
(01889) )
(01890) )
(01891) )
(01892) )
(01893) )
(01894) )
(01895) )
(01896) )
(01897) )
(01898) )
(01899) )
(01900) )
(01901) )
(01902) )
(01903) )
(01904) )
(01905) )
(01906) )
(01907) )
(01908) )
(01909) )
(01910) )
(01911) )
(01912) )
(01913) )
(01914) )
(01915) )
(01916) )
(01917) )
(01918) )
(01919) )
(01920) )
(01921) )
(01922) )
(01923) )
(01924) )
(01925) )
(01926) )
(01927) )
(01928) )
(01929) )
(01930) )
(01931) )
(01932) )
(01933) )
(01934) )
(01935) )
(01936) )
(01937) )
(01938) )
(01939) )
(01940) )
(01941) )
(01942) )
(01943) )
(01944) )
(01945) )
(01946) )
(01947) )
(01948) )
(01949) )
(01950) )
(01951) )
(01952) )
(01953) )
(01954) )
(01955) )
(01956) )
(01957) )
(01958) )
(01959) )
(01960) )
(01961) )
(01962) )
(01963) )
(01964) )
(01965) )
(01966) )
(01967) )
(01968) )
(01969) )
(01970) )
(01971) )
(01972) )
(01973) )
(01974) )
(01975) )
(01976) )
(01977) )
(01978) )
(01979) )
(01980) )
(01981) )
(01982) )
(01983) )
(01984) )
(01985) )
(01986) )
(01987) )
(01988) )
(01989) )
(01990) )
(01991) )
(01992) )
(01993) )
(01994) )
(01995) )
(01996) )
(01997) )
(01998) )
(01999) )
(02000) )
(02001) )
(02002) )
(02003) )
(02004) )
(02005) )
(02006) )
(02007) )
(02008) )
(02009) )
(02010) )
(02011) )
(02012) )
(02013) )
(02014) )
(02015) )
(02016) )
(02017) )
(02018) )
(02019) )
(02020) )
(02021) )
(02022) )
(02023) )
(02024) )
(02025) )
(02026) )
(02027) )
(02028) )
(02029) )
(02030) )
(02031) )
(02032) )
(02033) )
(02034) )
(02035) )
(02036) )
(02037) )
(02038) )
(02039) )
(02040) )
(02041) )
(02042) )
(02043) )
(02044) )
(02045) )
(02046) )
(02047) )
(02048) )
(02049) )
(02050) )
(02051) )
(02052) )
(02053) )
(02054) )
(02055) )
(02056) )
(02057) )
(02058) )
(02059) )
(02060) )
(02061) )
(02062) )
(02063) )
(02064) )
(02065) )
(02066) )
(02067) )
(02068) )
(02069) )
(02070) )
(02071) )
(02072) )
(02073) )
(02074) )
(02075) )
(02076) )
(02077) )
(02078) )
(02079) )
(02080) )
(02081) )
(02082) )
(02083) )
(02084) )
(02085) )
(02086) )
(02087) )
(02088) )
(02089) )
(02090) )
(02091) )
(02092) )
(02093) )
(02094) )
(02095) )
(02096) )
(02097) )
(02098) )
(02099) )
(02100) )
(02101) )
(02102) )
(02103) )
(02104) )
(02105) )
(02106) )
(02107) )
(02108) )
(02109) )
(02110) )
(02111) )
(02112) )
(02113) )
(02114) )
(02115) )
(02116) )
(02117) )
(02118) )
(02119) )
(02120) )
(02121) )
(02122) )
(02123) )
(02124) )
(02125) )
(02126) )
(02127) )
(02128) )
(02129) )
(02130) )
(02131) )
(02132) )
(02133) )
(02134) )
(02135) )
(02136) )
(02137) )
(02138) )
(02139) )
(02140) )
(02141) )
(02142) )
(02143) )
(02144) )
(02145) )
(02146) )
(02147) )
(02148) )
(02149) )
(02150) )
(02151) )
(02152) )
(02153) )
(02154) )
(02155) )
(02156) )
(02157) )
(02158) )
(02159) )
(02160) )
(02161) )
(02162) )
(02163) )
(02164) )
(02165) )
(02166) )
(02167) )
(02168) )
(02169) )
(02170) )
(02171) )
(02172) )
(02173) )
(02174) )
(02175) )
(02176) )
(02177) )
(02178) )
(02179) )
(02180) )
(02181) )
(02182) )
(02183) )
(02184) )
(02185) )
(02186) )
(02187) )
(02188) )
(02189) )
(02190) )
(02191) )
(02192) )
(02193) )
(02194) )
(02195) )
(02196) )
(02197) )
(02198) )
(02199) )
(02200) )
(02201) )
(02202) )
(02203) )
(02204) )
(02205) )
(02206) )
(02207) )
(02208) )
(02209) )
(02210) )
(02211) )
(02212) )
(02213) )
(02214) )
(02215) )
(02216) )
(02217) )
(02218) )
(02219) )
(02220) )
(02221) )
(02222) )
(02223) )
(02224) )
(02225) )
(02226) )
(02227) )
(02228) )
(02229) )
(02230) )
(02231) )
(02232) )
(02233) )
(02234) )
(02235) )
(02236) )
(02237) )
(02238) )
(02239) )
(02240) )
(02241) )
(02242) )
(02243) )
(02244) )
(02245) )
(02246) )
(02247) )
(02248) )
(02249) )
(02250) )
(02251) )
(02252) )
(02253) )
(02254) )
(02255) )
(02256) )
(02257) )
(02258) )
(02259) )
(02260) )
(02261) )
(02262) )
(02263) )
(02264) )
(02265) )
(02266) )
(02267) )
(02268) )
(02269) )
(02270) )
(02271) )
(02272) )
(02273) )
(02274) )
(02275) )
(02276) )
(02277) )
(02278) )
(02279) )
(02280) )
(02281) )
(02282) )
(02283) )
(02284) )
(02285) )
(02286) )
(02287) )
(02288) )
(02289) )
(02290) )
(02291) )
(02292) )
(02293) )
(02294) )
(02295) )
(02296) )
(02297) )
(02298) )
(02299) )
(02300) )
(02301) )
(02302) )
(02303) )
(02304) )
(02305) )
(02306) )
(02307) )
(02308) )
(02309) )
(02310) )
(02311) )
(02312) )
(02313) )
(02314) )
(02315) )
(02316) )
(02317) )
(02318) )
(02319) )
(02320) )
(02321) )
(02322) )
(02323) )
(02324) )
(02325) )
(02326) )
(02327) )
(02328) )
(02329) )
(02330) )
(02331) )
(02332) )
(02333) )
(02334) )
(02335) )
(02336) )
(02337) )
(02338) )
(02339) )
(02340) )
(02341) )
(02342) )
(02343) )
(02344) )
(02345) )
(02346) )
(02347) )
(02348) )
(02349) )
(02350) )
(02351) )
(02352) )
(02353) )
(02354) )
(02355) )
(02356) )
(02357) )
(02358) )
(02359) )
(02360) )
(02361) )
(02362) )
(02363) )
(02364) )
(02365) )
(02366) )
(02367) )
(02368) )
(02369) )
(02370) )
(02371) )
(02372) )
(02373) )
(02374) )
(02375) )
(02376) )
(02377) )
(02378) )
(02379) )
(02380) )
(02381) )
(02382) )
(02383) )
(02384) )
(02385) )
(02386) )
(02387) )
(02388) )
(02389) )
(02390) )
(02391) )
(02392) )
(02393) )
(02394) )
(02395) )
(02396) )
(02397) )
(02398) )
(02399) )
(02400) )
(02401) )
(02402) )
(02403) )
(02404) )
(02405) )
(02406) )
(02407) )
(02408) )
(02409) )
(02410) )
(02411) )
(02412) )
(02413) )
(02414) )
(02415) )
(02416) )
(02417) )
(02418) )
(02419) )
(02420) )
(02421) )
(02422) )
(02423) )
(02424) )
(02425) )
(02426) )
(02427) )
(02428) )
(02429) )
(02430) )
(02431) )
(02432) )
(02433) )
(02434) )
(02435) )
(02436) )
(02437) )
(02438) )
(02439) )
(02440) )
(02441) )
(02442) )
(02443) )
(02444) )
(02445) )
(02446) )
(02447) )
(02448) )
(02449) )
(02450) )
(02451) )
(02452) )
(02453) )
(02454) )
(02455) )
(02456) )
(02457) )
(02458) )
(02459) )
(02460) )
(02461) )
(02462) )
(02463) )
(02464) )
(02465) )
(02466) )
(02467) )
(02468) )
(02469) )
(02470) )
(02471) )
(02472) )
(02473) )
(02474) )
(02475) )
(02476) )
(02477) )
(02478) )
(02479) )
(02480) )
(02481) )
(02482) )
(02483) )
(02484) )
(02485) )
(02486) )
(02487) )
(02488) )
(02489) )
(02490) )
(02491) )
(02492) )
(02493) )
(02494) )
(02495) )
(02496) )
(02497) )
(02498) )
(02499) )
(02500) )
(02501) )
(02502) )
(02503) )
(02504) )
(02505) )
(02506) )
(02507) )
(02508) )
(02509) )
(02510) )
(02511) )
(02512) )
(02513) )
(02514) )
(02515) )
(02516) )
(02517) )
(02518) )
(02519) )
(02520) )
(02521) )
(02522) )
(02523) )
(02524) )
(02525) )
(02526) )
(02527) )
(02528) )
(02529) )
(02530) )
(02531) )
(02532) )
(02533) )
(02534) )
(02535) )
(02536) )
(02537) )
(02538) )
(02539) )
(02540) )
(02541) )
(02542) )
(02543) )
(02544) )
(02545) )
(02546) )
(02547) )
(02548) )
(02549) )
(02550) )
(02551) )
(02552) )
(02553) )
(02554) )
(02555) )
(02556) )
(02557) )
(02558) )
(02559) )
(02560) )
(02561) )
(02562) )
(02563) )
(02564) )
(02565) )
(02566) )
(02567) )
(02568) )
(02569) )
(02570) )
(02571) )
(02572) )
(02573) )
(02574) )
(02575) )
(02576) )
(02577) )
(02578) )
(02579) )
(02580) )
(02581) )
(02582) )
(02583) )
(02584) )
(02585) )
(02586) )
(02587) )
(02588) )
(02589) )
(02590) )
(02591) )
(02592) )
(02593) )
(02594) )
(02595) )
(02596) )
(02597) )
(02598) )
(02599) )
(02600) )
(02601) )
(02602) )
(02603) )
(02604) )
(02605) )
(02606) )
(02607) )
(02608) )
(02609) )
(02610) )
(02611) )
(02612) )
(02613) )
(02614) )
(02615) )
(02616) )
(02617) )
(02618) )
(02619) )
(02620) )
(02621) )
(02622) )
(02623) )
(02624) )
(02625) )
(02626) )
(02627) )
(02628) )
(02629) )
(02630) )
(02631) )
(02632) )
(02633) )
(02634) )
(02635) )
(02636) )
(02637) )
(02638) )
(02639) )
(02640) )
(02641) )
(02642) )
(02643) )
(02644) )
(02645) )
(02646) )
(02647) )
(02648) )
(02649) )
(02650) )
(02651) )
(02652) )
(02653) )
(02654) )
(02655) )
(02656) )
(02657) )
(02658) )
(02659) )
(02660) )
(02661) )
(02662) )
(02663) )
(02664) )
(02665) )
(02666) )
(02667) )
(02668) )
(02669) )
(02670) )
(02671) )
(02672) )
(02673) )
(02674) )
(02675) )
(02676) )
(02677) )
(02678) )
(02679) )
(02680) )
(02681) )
(02682) )
(02683) )
(02684) )
(02685) )
(02686) )
(02687) )
(02688) )
(02689) )
(02690) )
(02691) )
(02692) )
(02693) )
(02694) )
(02695) )
(02696) )
(026
```

PAGE 391

(BOUND)C0CA1G00016U.N50.2, 29-DEC-68 15:44:23, FDI WOLF
CORPAR(AB): UNATSTREIN, ERROR-CORRECT, AND DECODE PARAMETERS

03E26 C62A9B70 (01440)	PUSHMIL R2,DVLC(R5)	DECODE AND STORE GAIN
03E28 C62B9B70 (01449)	PUSHMIL R2,DVLC(R4)	DECODE AND STORE DCI
03E2A 06003E20 (01451)	CALL R0,GET74	GET DC2, DC3
03E2C 506F0006 (01452)	MOVNR R6,R7,C10	SAVE DC3 IN R6
03E2E 3C72 (01453)	LRS R7,2	SHIFT DC2 INTO POSITION
03E2F C62F9B70 (01454)	PUSHMIL R2,DVLC(R7)	DECODE AND STORE DC2
03E31 C62C9B70 (01455)	PUSHMIL R2,DVLC(R6)	DECODE AND STORE DC3
03E33 06003E20 (01457)	CALL R0,GET74	GET NI 4 BITS OF PITCH
03E35 3A73 (01458)	LLS R7,3	SHIFT INTO FINAL POSITION
03E36 405F (01459)	MOVNR R5,R7	AND SAVE IN R5
03E37 06003E20 (01460)	CALL R0,GET1	GET R0 C1
03E39 404F (01461)	MOVNR R4,R7	SAVE IT IN R4
03E3A 06003E20 (01462)	CALL R0,GET74	GET R2 C1, LO 3 BITS OF PITCH
03E3C 505F000E (01463)	TORNR R5,R7,C210	MOVE LO 3 OF PITCH IN WITH NI 4
03E3E 9C50001C (01464)	ADDR R5,142	ADD 14 TO "DECODE" (LSHIFTED 1, SO 2*)
03E40 3A56 (01465)	LLS R5,7-1	MAKE IT INTO UNIFORM FLT PT IN R5,R6
03E41 90600042 (01466)	MOVNR R6,42	EXPONENT
03E43 06540000 (01467)	MOVNR R5,0(R2)	STORE R5,R6
03E45 2022 (01468)	TORNR R2,2	STEP OUTPUT POINTER
03E46 3C71 (01469)	LRS R7,1	SHIFT R2 OF C1 INTO R2 POSITION
03E47 504F0000 (01470)	TORNR R4,R7,C2	R4 NOW HAS R2, R6 OF C1
03E49 06003E20 (01471)	CALL R0,GET1	GET R0 OF C2
03E4B 403F (01472)	MOVNR R3,R7	SAVE IT IN R3
03E4C 06003E20 (01473)	CALL R0,GET74	GET R221 OF C2, R1 OF C1
03E4E 503F001C (01475)	TORNR R3,R7,C321	ADD R221 OF C2 TO R0
03E50 4C7F (01477)	ADDR R7,R7	LEFTSHIFT TO POSITION R1 OF C1
03E51 504F0004 (01478)	TORNR R4,R7,C1	OR IT IN WITH R2, R6 OF C1
03E53 C62B9A00 (01479)	PUSHMIL R2,DVLC(R4)	DECODE AND STORE C1
03E55 C6269A10 (01480)	PUSHMIL R2,DVLC(R3)	DECODE AND STORE C2
03E57 06003E20 (01481)	CALL R0,GET1	GET R0 OF C3
03E59 405F (01482)	MOVNR R5,R7	SAVE IT IN R5
03E5A 06003E20 (01483)	CALL R0,GET1	GET R0 K1
03E5C 404E (01484)	MOVNR R4,R7	SAVE IT IN R4
03E5D 06003E20 (01485)	CALL R0,GET74	GET R54 K1, R21 C3
03E5F 4C7E (01486)	ADDR R7,R7	LEFTSHIFT TO POSITION R21 OF C3
03E60 505F000C (01487)	TORNR R5,R7,C21	PUT R21 C3 IN WITH R0
03E62 C62A9A00 (01488)	PUSHMIL R2,DVLC(R5)	DECODE AND STORE C3
03E64 4C7E (01489)	ADDR R7,R7	LEFTSHIFT AGAIN TO POSITION R54 OF K1
03E65 504F0000 (01490)	TORNR R4,R7,C54	SAVE THEM IN R4 WITH R0
03E67 06003E20 (01491)	CALL R0,GET1	GET R0 R2
03E69 405F (01492)	MOVNR R5,R7	STASH IT IN R5
03E6A 06003E20 (01493)	CALL R0,GET74	GET 4 MSB'S OF R2
03E6C 4C7E (01494)	ADDR R7,R7	LEFT SHIFT 1 TO POSITION THEM
03E6D 405E (01495)	TORNR R5,R7	ADD R4321 K2 TO R0 IN R5

(01501)		CALL	R0,C7T1	JGET R0 OF K3	
(01502)	060032D6	MOVW	R3,R7	JSTASH IT IN R3	
(01503)	403E				
(01504)					
(01505)	060032A0	CALL	R0,CET74	JGET R3 K3, B321 K1	
(01506)	563F0010	JORR	R3,R7,C3	JPUT R3 K3 IN R3 ALONG WITH R0	
(01507)	4C7E	ADDR	R7,R7	JLEFT SHIFT 1 TO POSITION B321 K1	
(01508)	063C76	JORR	R4,R7,C321	JPUT B321 K1 IN R4 WITH B54 AND R0	
(01509)	C2A09A30	PUSHMIL	R2,DULK1(R4)	JDECODE AND STORE K1	
(01510)	C62A9A80	PUSHMIL	R2,DULK2(R5)	JDECODE AND STORE K2	
(01511)					
(01512)	060032A0	CALL	R0,CET74	JGET R32 K4, B21 K3	
(01513)	50520010	MOVW	R5,R7,C32	JSAVE B32 K4 IN R5	
(01514)	4C7F	ADDR	R7,R7	JLEFTSHIFT 1 TO POSITION B21 K3	
(01515)	563E000C	JORR	R3,R7,C21	JPUT B21 K3 IN R3 ALONG WITH R3 AND R0	
(01516)	C6269A70	PUSHMIL	R2,DULK3(R3)	JDECODE AND STORE K3	
(01517)					
(01518)	060032CC	CALL	R0,CET2	JGET R10 K4	
(01519)	505F0006	JORR	R5,R7,C10	JPUT THEM IN R5 ALONG WITH B32	
(01520)	C62A9A10	PUSHMIL	R2,DULK4(R5)	JDECODE AND STORE K4	
(01521)					
(01522)	060032CC	CALL	R0,CET2	JGET R10 K5	
(01523)	405E	MOVW	R5,R7	JSAVE IT IN R5	
(01524)	060032CC	CALL	R0,CET2	JGET R10 K6	
(01525)	404F	MOVW	R4,R7	JSAVE IT IN R4	
(01526)	060032A0	CALL	R0,CET74	JGET R32 K6, B32 K5	
(01527)	504E0010	JORR	R4,R7,C32	JPUT R32 K6 IN R4 ALONG WITH R10 K6	
(01528)	3A72	LLS	R7,2	JLEFTSHIFT 2 TO POSITION B32 K5	
(01529)	565F0010	JORR	R5,R7,C32	JPUT R32 K5 IN R5 ALONG WITH R10 K5	
(01530)	C62A9B30	PUSHMIL	R2,DULK5(R5)	JDECODE AND STORE K5	
(01531)	C62A9B50	PUSHMIL	R2,DULK6(R4)	JDECODE AND STORE K6	
(01532)		MOVW	CLR+CI,SYSSFLCS		
(01533)	F000FFCE	RETURN			
(01534)	0C70				
(01535)					
(01536)					
(01537)					
(01538)					
(01539)					
(01540)					
(01541)					
(01542)					
(01543)					
(01544)					
(01545)					
(01546)					
(01547)					
(01548)					
(01549)	06C9F 0000	GET74:	POPR	R1,R6	JGET MODEN WORD INTO R6
(01550)	01EAF 0000		CVEN		
(01551)	001E02 507C0001	MOVW	R7,R6,1	JEXTRACT DATA BIT TO R7	
(01552)	001E04 4C7F	ADDR	R7,R7	JSHIFT LEFT 1 FASTER THAN LLS	
(01553)	001E05 301C	POPR	R1,R6		

03E0B 000E	(01607)	DATA 0'03'02	JRCVD 003
03E0C 0000	(01608)	DATA 0'00'02	JRCVD 004
03E0D 000A	(01609)	DATA 0'05'02	JRCVD 005
03E0E 001C	(01610)	DATA 0'16'02	JRCVD 006
03E0F 000E	(01611)	DATA 0'07'02	JRCVD 007
03E10 0000	(01612)	DATA 0'00'02	JRCVD 010
03E11 0012	(01613)	DATA 0'11'02	JRCVD 011
03E12 0004	(01614)	DATA 0'02'02	JRCVD 012
03E13 000E	(01615)	DATA 0'07'02	JRCVD 013
03E14 0000	(01616)	DATA 0'04'02	JRCVD 014
03E15 000F	(01617)	DATA 0'07'02	JRCVD 015
03E16 000E	(01618)	DATA 0'07'02	JRCVD 016
03E17 000E	(01619)	DATA 0'07'02	JRCVD 017
03E18 000F	(01620)	DATA 0'00'02	JRCVD 020
03E19 0012	(01621)	DATA 0'11'02	JRCVD 021
03E1A 001C	(01622)	DATA 0'16'02	JRCVD 022
03E1B 0016	(01623)	DATA 0'13'02	JRCVD 023
03E1C 001C	(01624)	DATA 0'16'02	JRCVD 024
03E1D 001A	(01625)	DATA 0'15'02	JRCVD 025
03E1E 001C	(01626)	DATA 0'16'02	JRCVD 026
03E1F 001C	(01627)	DATA 0'16'02	JRCVD 027
03E20 0012	(01628)	DATA 0'11'02	JRCVD 030
03E21 0012	(01629)	DATA 0'11'02	JRCVD 031
03E22 0014	(01630)	DATA 0'12'02	JRCVD 032
03E23 0012	(01631)	DATA 0'11'02	JRCVD 033
03E24 0010	(01632)	DATA 0'14'02	JRCVD 034
03E25 0012	(01633)	DATA 0'11'02	JRCVD 035
03E26 001C	(01634)	DATA 0'16'02	JRCVD 036
03E27 000F	(01635)	DATA 0'07'02	JRCVD 037
03E28 0000	(01636)	DATA 0'00'02	JRCVD 040
03E29 000A	(01637)	DATA 0'05'02	JRCVD 041
03E2A 0004	(01638)	DATA 0'02'02	JRCVD 042
03E2B 0016	(01639)	DATA 0'13'02	JRCVD 043
03E2C 000A	(01640)	DATA 0'05'02	JRCVD 044
03E2D 000A	(01641)	DATA 0'05'02	JRCVD 045
03E2E 000C	(01642)	DATA 0'06'02	JRCVD 046
03E2F 000A	(01643)	DATA 0'05'02	JRCVD 047
03E30 0004	(01644)	DATA 0'02'02	JRCVD 050
03E31 0002	(01645)	DATA 0'01'02	JRCVD 051
03E32 0004	(01646)	DATA 0'02'02	JRCVD 052
03E33 0004	(01647)	DATA 0'02'02	JRCVD 053
03E34 0010	(01648)	DATA 0'14'02	JRCVD 054
03E35 000A	(01649)	DATA 0'05'02	JRCVD 055
03E36 0004	(01650)	DATA 0'02'02	JRCVD 056
03E37 000E	(01651)	DATA 0'07'02	JRCVD 057
03E38 0010	(01652)	DATA 0'10'02	JRCVD 060
03E39 0016	(01653)	DATA 0'13'02	JRCVD 061
03E3A 0016	(01654)	DATA 0'13'02	JRCVD 062
03E3B 0016	(01655)	DATA 0'13'02	JRCVD 063
03E3C 0010	(01656)	DATA 0'14'02	JRCVD 064
03E3D 000A	(01657)	DATA 0'05'02	JRCVD 065
03E3E 001C	(01658)	DATA 0'16'02	JRCVD 066
03E3F 0016	(01659)	DATA 0'13'02	JRCVD 067

03F10 0018	(01660)	DATA 0'14'02	JRCVD 070
03F11 0012	(01661)	DATA 0'11'02	JRCVD 071
03F12 0004	(01662)	DATA 0'02'02	JRCVD 072
03F13 0016	(01663)	DATA 0'13'02	JRCVD 073
03F14 0010	(01664)	DATA 0'14'02	JRCVD 074
03F15 0018	(01665)	DATA 0'14'02	JRCVD 075
03F16 0010	(01666)	DATA 0'14'02	JRCVD 076
03F17 0012	(01667)	DATA 0'17'02	JRCVD 077
03F18 0008	(01668)	DATA 0'08'02	JRCVD 100
03F19 0006	(01669)	DATA 0'03'02	JRCVD 101
03F1A 0006	(01670)	DATA 0'03'02	JRCVD 102
03F1B 0006	(01671)	DATA 0'03'02	JRCVD 103
03F1C 0008	(01672)	DATA 0'04'02	JRCVD 104
03F1D 001A	(01673)	DATA 0'15'02	JRCVD 105
03F1E 000C	(01674)	DATA 0'06'02	JRCVD 106
03F1F 0006	(01675)	DATA 0'03'02	JRCVD 107
03F20 0008	(01676)	DATA 0'04'02	JRCVD 110
03F21 0002	(01677)	DATA 0'01'02	JRCVD 111
03F22 0014	(01678)	DATA 0'12'02	JRCVD 112
03F23 0006	(01679)	DATA 0'03'02	JRCVD 113
03F24 0008	(01680)	DATA 0'04'02	JRCVD 114
03F25 0008	(01681)	DATA 0'04'02	JRCVD 115
03F26 0008	(01682)	DATA 0'04'02	JRCVD 116
03F27 000E	(01683)	DATA 0'07'02	JRCVD 117
03F28 0010	(01684)	DATA 0'10'02	JRCVD 120
03F29 001A	(01685)	DATA 0'15'02	JRCVD 121
03F2A 0014	(01686)	DATA 0'12'02	JRCVD 122
03F2B 0006	(01687)	DATA 0'03'02	JRCVD 123
03F2C 001A	(01688)	DATA 0'15'02	JRCVD 124
03F2D 001A	(01689)	DATA 0'15'02	JRCVD 125
03F2E 001C	(01690)	DATA 0'16'02	JRCVD 126
03F2F 001A	(01691)	DATA 0'15'02	JRCVD 127
03F30 0014	(01692)	DATA 0'12'02	JRCVD 130
03F31 0012	(01693)	DATA 0'11'02	JRCVD 131
03F32 0014	(01694)	DATA 0'12'02	JRCVD 132
03F33 0014	(01695)	DATA 0'12'02	JRCVD 133
03F34 0008	(01696)	DATA 0'04'02	JRCVD 134
03F35 001A	(01697)	DATA 0'15'02	JRCVD 135
03F36 0014	(01698)	DATA 0'12'02	JRCVD 136
03F37 001E	(01699)	DATA 0'17'02	JRCVD 137
03F38 0010	(01700)	DATA 0'10'02	JRCVD 140
03F39 0002	(01701)	DATA 0'01'02	JRCVD 141
03F3A 000C	(01702)	DATA 0'06'02	JRCVD 142
03F3B 0006	(01703)	DATA 0'03'02	JRCVD 143
03F3C 000C	(01704)	DATA 0'06'02	JRCVD 144
03F3D 000A	(01705)	DATA 0'05'02	JRCVD 145
03F3E 000C	(01706)	DATA 0'06'02	JRCVD 146
03F3F 000C	(01707)	DATA 0'06'02	JRCVD 147
03F40 0002	(01708)	DATA 0'01'02	JRCVD 150
03F41 0002	(01709)	DATA 0'01'02	JRCVD 151
03F42 0004	(01710)	DATA 0'02'02	JRCVD 152
03F43 0002	(01711)	DATA 0'01'02	JRCVD 153
03F44 000A	(01712)	DATA 0'04'02	JRCVD 154

PAGE 44: (CONDICOCAL16>DDH16U.H50.2, 29-Dec-88 15:44:23, Ed: VULF
 COMPAR(16): UNBITSTREAM, ZERO8-CORRECT, AND DECODE PARAMETERS

03F45 0002	(01713)	DATA 0'01'02	JACVD 155
03F46 000C	(01714)	DATA 0'06'02	JACVD 156
03F47 001F	(01715)	DATA 0'17'02	JACVD 157
03F48 0010	(01716)	DATA 0'10'02	JACVD 160
03F49 0010	(01717)	DATA 0'10'02	JACVD 161
03F4A 0010	(01718)	DATA 0'10'02	JACVD 162
03F4B 0016	(01719)	DATA 0'13'02	JACVD 163
03F4C 0010	(01720)	DATA 0'10'02	JACVD 164
03F4D 001A	(01721)	DATA 0'15'02	JACVD 165
03F4E 000C	(01722)	DATA 0'06'02	JACVD 166
03F4F 001F	(01723)	DATA 0'17'02	JACVD 167
03F50 001A	(01724)	DATA 0'10'02	JACVD 170
03F51 0002	(01725)	DATA 0'01'02	JACVD 171
03F52 0014	(01726)	DATA 0'12'02	JACVD 172
03F53 001E	(01727)	DATA 0'17'02	JACVD 173
03F54 0010	(01728)	DATA 0'14'02	JACVD 174
03F55 001E	(01729)	DATA 0'17'02	JACVD 175
03F56 001F	(01730)	DATA 0'17'02	JACVD 176
03F57 001F	(01731)	DATA 0'17'02	JACVD 177
	(01732)		

PAGE 45: (ADDR)OCAL16>88H16U.M50.2, 29-Dec-88 15:44:23, 2di WOLF
APU3-DECODE()

```

(01733) APU3-DECODE()          DECODE RESIDUAL
(01734) J. WOLF, 10/21/86
(01735)
(01736) BINDS TO APU3-DECODE OR APU3-DECODE
(01737)
(01738) UNBITSTREAMS AND DECODES THE RECEIVED RESIDUAL BITS FROM AN "RBITSM"
(01739) BUFFER (RDATA OR RBIT) TO AN "RRESOURCE" BUFFER (RSRA OR RSRR).
(01740) RSRI(N) = DVLUF(RBIT(N+2),RBIT(N+1))
(01741) WE PULL IN 2 HALFWORDS FROM THE RBIT BUFFER, THE FIRST OF WHICH HAS
(01742) THE RESIDUAL LSB IN ITS RIGHTMOST BIT (R16) AND THE SECOND OF WHICH
(01743) HAS THE MSB. WE SHIFT THOSE BITS TO BIT 7, THEN XLSBT THEM
(01744) INTO T1 AND T2 RESPECTIVELY, THEN DO LOGIC ON THE T-BITS TO
(01745) SELECT THE PROPER OUTPUT VALUE FROM THE 4-LONG DVLUF TABLE.
(01746) DVLUF IS THE (LENGTH=4) DECODING TABLE FOR THE 2-BIT RESIDUAL DATA,
(01747) WHICH USES THE FOLDED BINARY CODE.
(01748)
(01749) INPUT SEQUENCE:
(01750) (2**9), DVLUF(0),...,DVLUF(3),
(01751) RBIT(1), RBIT(2), ..., RBIT(2*RESID), (P1)
(01752) OUTPUT SEQUENCE:
(01753) RSRI(0), RSRI(1), ..., RSRI(RESID-1), (EO)
(01754)
(01755) JOURNATION:
(01756) -10 APU CYCLES = -1.0 USEC PER RESIDUAL SAMPLE
(01757) 3 NEW REFS = -1.5 USEC (BUS 1) PER RESIDUAL SAMPLE
(01758)
(01759) APU REGISTER USAGE:
(01760) LEFT RIGHT
(01761) A0 DVLUF(0) -
(01762) A1 DVLUF(1) -
(01763) A2 DVLUF(2) -
(01764) A3 DVLUF(3) -
(01765) A4 SHIFTED LSB
(01766) M0: 2**9
(01767) M4: LSB WORD
(01768)
(01769) EVEN
(01770) DATA DCDUSSA JAPU START ADDRESS
(01771) DATA DCDUSSZ JAPU MODULE SIZE
(01772)
(01773) DCDU: BEGIN APU(DDCU)
(01774) DCDUSSA: SET(UN)
(01775) MOV(IQA,M0)
(01776) MOV(IQA,M0) \ NOP
(01777) MOV(IQA,M1) \ NOP
(01778) MOV(IQA,M2) \ NOP
(01779) MOV(IQA,M3) \ NOP
(01780) MOV(IQA,M4) \ NOP
(01781) MOV(IQA,M4) \ NOP
(01782) MUL(M0,M4)
(01783) MOV(P,M4)
(01784) JMAIN LOOP
(01785) B1: ALIGN(A4)
(01786)
(01787) JBY MULT. BY (2**9) AND THEN ALIGNING.

```

```

PAGE 46: [M001]DCAL16>B0016U.M00.2, 29-Dec-88 15:44:23, Ed: WOLF
          SPJ3-DECODE()
          DECODE RESIDUAL
          (PICK UP THE NEXT LSB)
          (MOVE THE DATA BITS INTO T1, T2)
          (PICK UP THE NEXT MSB)
          (START THE NEXT MULTIPLY)
          (WAIT FOR XLSBT TO FINISH)
          (JUMP IF MSB=1)
          (MSB=0) JUMP IF LSB=1
          (DATUM=00) RETURN DVALUF(0)

          (DATUM=01: DVALUF(1))

          (MSB=1) JUMP IF LSB=1
          (DATUM=10: DVALUF(2))

          (DATUM=11: DVALUF(3))

          (OUTPUT DECODED VALUE)
          (LOOP IF WE'VE NOT FINISHED OUTPUT)

          MOV(TQA,M4) \ NOP
          MOV(M4),XLSBT(A4)
          MOV \ MOV(TQA,M4)
          MUL(M4,M4)
          MOV(M4,M4)
          MOV(M4,M4)
          JUMPS(DCD$11,T1)
          DCD$0X: JUMPS(DCD$01,T1)
          DCD$00: R(A0) \ NOP
          JUMP(DCD$OUT)

          DCD$01: R(A1) \ NOP
          JUMP(DCD$OUT)

          DCD$11: JUMPS(DCD$11,T1)
          DCD$10: R(A2) \ NOP
          JUMP(DCD$OUT)

          DCD$11: R(A3) \ NOP
          DCD$0X: MOV(P,A4)
          MOV(R,00) \ NOP
          JUMPC(RI,ED)
          CLEAR(RA)
          JUMP(0)

          DCDU$SZ = 1A - DCDU$SA
          03F9B END

```

PAGE 47: (0000) <0C116> 29-DEC-88 15:44:23, Ed: WOLF
APS PROGRAM FOR DECODA

```

(01011) APS3-DECODA() APS PROGRAM FOR DECODA
(01012)
(01013) J. WOLF, 10/21/88
(01014)
(01015) BINDS TO APS3-DECOD
(01016)
(01017) APS PROGRAM FOR DECODING THE RESIDUAL FOR "A" BUFFERS (RSTA TO RSRA).
(01018) BECAUSE WE CAN'T SPARE BID'S FOR THE RSTA BUFFERS, WE HAVE 2
(01019) VERSIONS OF THE DECODE MODULE, ONE WITH (RSTA,RSRA) "HARD BOUND"
(01020) IN, AND ONE WITH (RSTA,RSRB).
(01021)
(01022) HEADER BLOCK
(01023)
(01024) EVEN
(01025) ADDR 0
(01026) ADDR 0
(01027) DATA 0
(01028) DATA DCDS$Z
(01029) ADDR 0
(01030) EVEN
(01031)
(01032) INPUT PROGRAM:
(01033) BR0: RSTA(M)
(01034) BR1: LOOP COUNTER (STARTS AT HRESID-1)
(01035)
(01036) DCDS$Z
(01037) DCDS$Z: BEGIN APS(DCDSA)
(01038) SET(R0)
(01039) LOAD(BR0,TWOH9(1),TF)
(01040) LOAD(BR0,DVLUF(1),TF)
(01041) ADD(BR0,$S,TF)
(01042) ADD(BR0,$S,TF)
(01043) ADD(BR0,$S,TF)
(01044) ADD(BR0,$S,TF)
(01045)
(01046) LOAD(BR0,RSTA(1))
(01047) LOAD(BR1,HRESID-1)
(01048) JMAIN LOOP
(01049) BR1:
(01050) ADD(BR0,$S,TF)
(01051) ADD(BR0,$S,TF)
(01052) SUBL(BR1,1),JUNPP(R1)
(01053) CLFAR(R1)
(01054) NOP(0)
(01055)
(01056) OUTPUT PROGRAM:
(01057) BR0: RSRA(M)
(01058) BR1: LOOP COUNTER (INIT TO HRESID-1)
(01059)
(01060) DCDS$Z: SET(RA)
(01061) LOAD(BR0,RSRA-$S(1))
(01062) LOAD(BR1,HRESID-1)
(01063) JMAIN LOOP
(01064) BR2:
(01065) ADD(RA0,$S,TF)
(01066)
(01067) JSTART APU
(01068) JINIT BR0 TO POINT TO OUTPUT BUFFER-$S
(01069) JINIT LOOP COUNTER
(01070) JGEN OUTPUT ADR

```

PAGE 40: CRRMOJ<0C41C>00016U.NSD.2, 29-Dec-88 15:44:23, Ed: NUL.F
 APS PROGRAM FOR DECODA

AP23-DECODA()

A12 03FCA 24111101 (01064) SWUL(0V1,1), JUMPP(02)

A13 03FC6 26200030 (01045) CLEAR(RO)

A14 03FC0 20000020 (01066) MUP(0)

03FCA 00000021 (01067) END

03FCA 00000021 (01068) DCDSASZ = 0L - DCDSAI

03FCA 4000 (01069) ?

03FCB 003F (01070) FROM9: DATA \$0000,\$003E

! COMPUTE BUS 1 MODULE SIZE

!(2***9) FOR DECODA, DECODB

```

PAGE
49: (DDDD)(<DCA16>88W16V.WSU.2, 29-DEC-88 15:44:23, Ed: VOLF
      APS3--RECDDB() APS PROGRAM FOR DECODE

```

[illegible]

PAGE 50: C00M01C0C1C000160.M50.2, 29-DEC-88 15:44:23, E41 WULF
APS3-DECODE() APS PROGRAM FOR DECODE

0377E (01924) ;
(01925)

END END OF FILE

PAGE 501 (0000100116>000160.W50.2, 29-Dec-88 1514123, 241 WULF
APS3-DECODE() APS PROGRAM FOR DECODE

0377E (01924) ; END OF FILE
(01925)

PAGE 511 (RUND)DC116>RUND16U.MSD.2, 29-Dec-80 15:44:23, Ed: WULF
AP53-DZCDB() APS PROGRAM FOR DECDB

ACQTRB:	0000 (00010) (00945)		
ADGCR:	040A2 (00454) (00476)		
ADGCPVA:	0490A (00563) (00600)		
ADGCPVB:	0490A (00564) (00615)		
ADGCRSC:	04906 (00570) (00596)		
ADGLQOP:	00004 (00460) (00472)		
ADSTW:	04902 (00575) (00590)	(00597)	
ADSTWOP:	00015 (00465) (00473)		
ADSSZ:	0002C (00452) (00476)		
ADSEWB:	01F1F (00437) (00464)		
ADAMINT:	0495C (00129) (00172)	(00570)	
ADSEWD:	01F77 (00430) (00471)		
ADPTN:	04952 (00561) (00579)		
ADCTBL:	00002 (00440) (00403)		
ADCTB:	00555 (00171) (00310)	(00314)	
ADPO:	00542 (00270) (00281)	(00571)	
ADPTORG:	00000 (00013) (00135)	(00140)	
ADMSWB:	03CA6 (01037) (01046)		
ADMSWB:	03CA2 (01033) (01042)	(01049)	
ADMSWB:	03C06 (00132) (00165)	(01040)	
ADMSWB:	00001 (00017) (00023)	(00024)	(01203) (01216) (01227) (01235) (01240) (01259)
ADMSWB:	00002 (00010) (00023)	(00024)	(00026)
ADMSWB:	00003 (00023) (01169)	(01242)	(01274) (01276) (01282) (01288)
ADMSWB:	00004 (00019) (00024)	(00025)	(00026)
ADMSWB:	00007 (00024) (01195)		
ADMSWB:	00000 (00020) (00025)	(00026)	(01207) (01266)
ADMSWB:	0000C (00025) (01273)	(01290)	(01293)
ADMSWB:	0000E (00026) (01221)	(01263)	
ADMSWB:	00010 (00021) (00027)		
ADMSWB:	00020 (00022) (00027)		
ADMSWB:	00030 (00027) (01230)		
ADMSWB:	07777 (00014) (00342)	(00343)	(00344) (00345) (00346) (00437) (00438) (00494) (00495)
ADMSWB:	03DA4 (01302) (01307)		
ADMSWB:	00002 (00029) (00035)	(00037)	
ADMSWB:	00004 (00030) (00035)	(00036)	(00039) (01470)
ADMSWB:	00006 (00035) (01445)	(01452)	(01519)
ADMSWB:	09CAC (00197) (00190)	(01202)	
ADMSWB:	00000 (00031) (00036)	(00037)	(00039) (01470)
ADMSWB:	0000C (00036) (01490)	(01515)	
ADMSWB:	0000E (00037) (01463)	(01579)	
ADMSWB:	09CS4 (00190) (00199)	(01215)	
ADMSWB:	00010 (00032) (00030)	(00039)	(01506) (01576)
ADMSWB:	00018 (00030) (01513)	(01527)	(01529)
ADMSWB:	0001C (00039) (01476)	(01500)	
ADMSWB:	09C64 (00199) (00200)	(01226)	
ADMSWB:	00020 (00033) (00040)		
ADMSWB:	00040 (00034) (00040)		
ADMSWB:	00060 (00040) (01493)		
ADMSWB:	00002 (00370) (00372)		
ADMSWB:	074C6 (00350) (00372)		
ADMSWB:	00000 (00370)		
ADMSWB:	00000 (00084)	(00590)	(00663) (00016) (01042) (01154) (01295) (01533)
ADMSWB:	03F12 (00126) (01434)		

CONTAB:	3300	(01572)	(01603)
CS909W08:	0217C	(00042)	(00130)
CTAC1:	0900	(00049)	(00050)
CTAC2:	0910	(00050)	(00052)
CTAC3:	0900	(00051)	
CTAC6:	0900	(00059)	(00060)
CTAC:	0970	(00050)	(00059)
CTW1:	0930	(00052)	(00053)
CTW2:	0900	(00053)	(00054)
CTW3:	0900	(00054)	(00055)
CTW4:	0910	(00055)	(00056)
CTW5:	0930	(00056)	(00057)
CTW6:	0950	(00057)	(00058)
CTW0:	0900	(00060)	(00061)
01631MT1:	0402	(00076)	(00149)
01631MT2:	0430	(00077)	(00156)
02281MT1:	0400	(00070)	(00163)
02331MT1:	041E	(00079)	(00170)
04800:	040E	(00510)	(00530)
048CPVA:	03C8	(01619)	(01052)
048CPVB:	0300	(01020)	(01059)
048LOOP:	0001	(00512)	(00526)
048STOP:	0001S	(00510)	(00527)
0432:	002C	(00500)	(00530)
04880B:	03E7	(00494)	(00517)
04880B:	0377	(00495)	(00525)
040PTA:	037C	(01020)	(01030)
040CTAL:	0002	(00497)	(00537)
040TA:	0050	(00164)	(00313)
040P:	00549	(00200)	(01039)
040S00:	00012	(01793)	(01796)
040S01:	00014	(01792)	
040S01:	00011	(01792)	
040S10:	00017	(01000)	
040S11:	00019	(01799)	(01003)
040S12:	00016	(01791)	(01799)
040S00T1:	0001A	(01794)	(01797)
040S1:	03A0	(00137)	(01036)
040S1S1:	002A	(01020)	(01060)
040S11:	0300	(01037)	(01060)
040S10:	0000E	(01030)	(01059)
040S0:	0304	(00142)	(01091)
040S0S2:	002A	(01003)	(01923)
040S01:	0304	(01092)	(01923)
040S00:	0000E	(01093)	(01914)
040S:	035A	(00136)	(00141)
040S0S1:	0000	(01770)	(01774)
040S0S2:	00017	(01771)	(01009)
040S1T:	05C40	(00196)	(00197)
040C1:	0910	(00061)	(00062)
040C2:	0910	(00062)	(00064)
040C3:	0910	(00063)	(01491)
040C:	0900	(00071)	(00072)
040S0S1:	0000	(01770)	(01774)
040S0S2:	00017	(01771)	(01009)
040S1T:	05C40	(00196)	(00197)
040C1:	0910	(00061)	(00062)
040C2:	0910	(00062)	(00064)
040C3:	0910	(00063)	(01491)
040C:	0900	(00071)	(00072)
040S0S1:	0000	(01770)	(01774)
040S0S2:	00017	(01771)	(01009)
040S1T:	05C40	(00196)	(00197)
040C1:	0910	(00061)	(00062)
040C2:	0910	(00062)	(00064)
040C3:	0910	(00063)	(01491)
040C:	0900	(00071)	(00072)
040S0S1:	0000	(01770)	(01774)
040S0S2:	00017	(01771)	(01009)
040S1T:	05C40	(00196)	(00197)
040C1:	0910	(00061)	(00062)
040C2:	0910	(00062)	(00064)
040C3:	0910	(00063)	(01491)
040C:	0900	(00071)	(00072)
040S0S1:	0000	(01770)	(01774)
040S0S2:	00017	(01771)	(01009)
040S1T:	05C40	(00196)	(00197)
040C1:	0910	(00061)	(00062)
040C2:	0910	(00062)	(00064)
040C3:	0910	(00063)	(01491)
040C:	0900	(00071)	(00072)
040S0S1:	0000	(01770)	(01774)
040S0S2:	00017	(01771)	(01009)
040S1T:	05C40	(00196)	(00197)
040C1:	0910	(00061)	(00062)
040C2:	0910	(00062)	(00064)
040C3:	0910	(00063)	(01491)
040C:	0900	(00071)	(00072)
040S0S1:	0000	(01770)	(01774)
040S0S2:	00017	(01771)	(01009)
040S1T:	05C40	(00196)	(00197)
040C1:	0910	(00061)	(00062)
040C2:	0910	(00062)	(00064)
040C3:	0910	(00063)	(01491)
040C:	0900	(000	

PAGE 53: [R0ND]C0C116>R0N16W-MS0-2, 29-Dec-80 15:44:23, ED: WOLF
AP33-RECORD() APS PROGRAM FOR RECORD

09A70	(00070)	(00071)	(01440)
09A71	(00064)	(00065)	(01509)
09A72	(00065)	(00066)	(01510)
09A73	(00066)	(00067)	(01516)
09A74	(00067)	(00068)	(01520)
09A75	(00068)	(00069)	(01530)
09A76	(00069)	(00070)	(01531)
09A77	(00070)	(00071)	(01531)
09A78	(00071)	(00072)	(01531)
09A79	(00072)	(00073)	(01531)
09A80	(00073)	(00074)	(01531)
09A81	(00074)	(00075)	(01531)
09A82	(00075)	(00076)	(01531)
09A83	(00076)	(00077)	(01531)
09A84	(00077)	(00078)	(01531)
09A85	(00078)	(00079)	(01531)
09A86	(00079)	(00080)	(01531)
09A87	(00080)	(00081)	(01531)
09A88	(00081)	(00082)	(01531)
09A89	(00082)	(00083)	(01531)
09A90	(00083)	(00084)	(01531)
09A91	(00084)	(00085)	(01531)
09A92	(00085)	(00086)	(01531)
09A93	(00086)	(00087)	(01531)
09A94	(00087)	(00088)	(01531)
09A95	(00088)	(00089)	(01531)
09A96	(00089)	(00090)	(01531)
09A97	(00090)	(00091)	(01531)
09A98	(00091)	(00092)	(01531)
09A99	(00092)	(00093)	(01531)
09B00	(00093)	(00094)	(01531)
09B01	(00094)	(00095)	(01531)
09B02	(00095)	(00096)	(01531)
09B03	(00096)	(00097)	(01531)
09B04	(00097)	(00098)	(01531)
09B05	(00098)	(00099)	(01531)
09B06	(00099)	(00100)	(01531)
09B07	(00100)	(00101)	(01531)
09B08	(00101)	(00102)	(01531)
09B09	(00102)	(00103)	(01531)
09B10	(00103)	(00104)	(01531)
09B11	(00104)	(00105)	(01531)
09B12	(00105)	(00106)	(01531)
09B13	(00106)	(00107)	(01531)
09B14	(00107)	(00108)	(01531)
09B15	(00108)	(00109)	(01531)
09B16	(00109)	(00110)	(01531)
09B17	(00110)	(00111)	(01531)
09B18	(00111)	(00112)	(01531)
09B19	(00112)	(00113)	(01531)
09B20	(00113)	(00114)	(01531)
09B21	(00114)	(00115)	(01531)
09B22	(00115)	(00116)	(01531)
09B23	(00116)	(00117)	(01531)
09B24	(00117)	(00118)	(01531)
09B25	(00118)	(00119)	(01531)
09B26	(00119)	(00120)	(01531)
09B27	(00120)	(00121)	(01531)
09B28	(00121)	(00122)	(01531)
09B29	(00122)	(00123)	(01531)
09B30	(00123)	(00124)	(01531)
09B31	(00124)	(00125)	(01531)
09B32	(00125)	(00126)	(01531)
09B33	(00126)	(00127)	(01531)
09B34	(00127)	(00128)	(01531)
09B35	(00128)	(00129)	(01531)
09B36	(00129)	(00130)	(01531)
09B37	(00130)	(00131)	(01531)
09B38	(00131)	(00132)	(01531)
09B39	(00132)	(00133)	(01531)
09B40	(00133)	(00134)	(01531)
09B41	(00134)	(00135)	(01531)
09B42	(00135)	(00136)	(01531)
09B43	(00136)	(00137)	(01531)
09B44	(00137)	(00138)	(01531)
09B45	(00138)	(00139)	(01531)
09B46	(00139)	(00140)	(01531)
09B47	(00140)	(00141)	(01531)
09B48	(00141)	(00142)	(01531)
09B49	(00142)	(00143)	(01531)
09B50	(00143)	(00144)	(01531)
09B51	(00144)	(00145)	(01531)
09B52	(00145)	(00146)	(01531)
09B53	(00146)	(00147)	(01531)
09B54	(00147)	(00148)	(01531)
09B55	(00148)	(00149)	(01531)
09B56	(00149)	(00150)	(01531)
09B57	(00150)	(00151)	(01531)
09B58	(00151)	(00152)	(01531)
09B59	(00152)	(00153)	(01531)
09B60	(00153)	(00154)	(01531)
09B61	(00154)	(00155)	(01531)
09B62	(00155)	(00156)	(01531)
09B63	(00156)	(00157)	(01531)
09B64	(00157)	(00158)	(01531)
09B65	(00158)	(00159)	(01531)
09B66	(00159)	(00160)	(01531)
09B67	(00160)	(00161)	(01531)
09B68	(00161)	(00162)	(01531)
09B69	(00162)	(00163)	(01531)
09B70	(00163)	(00164)	(01531)
09B71	(00164)	(00165)	(01531)
09B72	(00165)	(00166)	(01531)
09B73	(00166)	(00167)	(01531)
09B74	(00167)	(00168)	(01531)
09B75	(00168)	(00169)	(01531)
09B76	(00169)	(00170)	(01531)
09B77	(00170)	(00171)	(01531)
09B78	(00171)	(00172)	(01531)
09B79	(00172)	(00173)	(01531)
09B80	(00173)	(00174)	(01531)
09B81	(00174)	(00175)	(01531)
09B82	(00175)	(00176)	(01531)
09B83	(00176)	(00177)	(01531)
09B84	(00177)	(00178)	(01531)
09B85	(00178)	(00179)	(01531)
09B86	(00179)	(00180)	(01531)
09B87	(00180)	(00181)	(01531)
09B88	(00181)	(00182)	(01531)
09B89	(00182)	(00183)	(01531)
09B90	(00183)	(00184)	(01531)
09B91	(00184)	(00185)	(01531)
09B92	(00185)	(00186)	(01531)
09B93	(00186)	(00187)	(01531)
09B94	(00187)	(00188)	(01531)
09B95	(00188)	(00189)	(01531)
09B96	(00189)	(00190)	(01531)
09B97	(00190)	(00191)	(01531)
09B98	(00191)	(00192)	(01531)
09B99	(00192)	(00193)	(01531)
09C00	(00193)	(00194)	(01531)
09C01	(00194)	(00195)	(01531)
09C02	(00195)	(00196)	(01531)
09C03	(00196)	(00197)	(01531)
09C04	(00197)	(00198)	(01531)
09C05	(00198)	(00199)	(01531)
09C06	(00199)	(00200)	(01531)
09C07	(00200)	(00201)	(01531)
09C08	(00201)	(00202)	(01531)
09C09	(00202)	(00203)	(01531)
09C10	(00203)	(00204)	(01531)
09C11	(00204)	(00205)	(01531)
09C12	(00205)	(00206)	(01531)
09C13	(00206)	(00207)	(01531)
09C14	(00207)	(00208)	(01531)
09C15	(00208)	(00209)	(01531)
09C16	(00209)	(00210)	(01531)
09C17	(00210)	(00211)	(01531)
09C18	(00211)	(00212)	(01531)
09C19	(00212)	(00213)	(01531)
09C20	(00213)	(00214)	(01531)
09C21	(00214)	(00215)	(01531)
09C22	(00215)	(00216)	(01531)
09C23	(00216)	(00217)	(01531)
09C24	(00217)	(00218)	(01531)
09C25	(00218)	(00219)	(01531)
09C26	(00219)	(00220)	(01531)
09C27	(00220)	(00221)	(01531)
09C28	(00221)	(00222)	(01531)
09C29	(00222)	(00223)	(01531)
09C30	(00223)	(00224)	(01531)
09C31	(00224)	(00225)	(01531)
09C32	(00225)	(00226)	(01531)
09C33	(00226)	(00227)	(01531)
09C34	(00227)	(00228)	(01531)
09C35	(00228)	(00229)	(01531)
09C36	(00229)	(00230)	(01531)
09C37	(00230)	(00231)	(01531)
09C38	(00231)	(00232)	(01531)
09C39	(00232)	(00233)	(01531)
09C40	(00233)	(00234)	(01531)
09C41	(00234)	(00235)	(01531)
09C42	(00235)	(00236)	(01531)
09C43	(00236)	(00237)	(01531)
09C44	(00237)	(00238)	(01531)
09C45	(00238)	(00239)	(01531)
09C46	(00239)	(00240)	(01531)
09C47	(00240)	(00241)	(01531)
09C48	(00241)	(00242)	(01531)
09C49	(00242)	(00243)	(01531)
09C50	(00243)	(00244)	(01531)
09C51	(00244)	(00245)	(01531)
09C52	(00245)	(00246)	(01531)
09C53	(00246)	(00247)	(01531)
09C54	(00247)	(00248)	(01531)
09C55	(00248)	(00249)	(01531)
09C56	(00249)	(00250)	(01531)
09C57	(00250)	(00251)	(01531)
09C58	(00251)	(00252)	(01531)
09C59	(00252)	(00253)	(01531)
09C60	(00253)	(00254)	(01531)
09C61	(00254)	(00255)	(01531)
09C62	(00255)	(00256)	(01531)
09C63	(00256)	(00257)	(01531)
09C64	(00257)	(00258)	(01531)
09C65	(00258)	(00259)	(01531)
09C66	(00259)	(00260)	(01531)
09C67	(00260)	(00261)	(01531)
09C68	(00261)	(00262)	(01531)
09C69	(00262)	(00263)	(01531)
09C70	(00263)	(00264)	(01531)
09C71	(00264)	(00265)	(01531)
09C72	(00265)	(00266)	(01531)
09C73	(00266)	(00267)	(01531)
09C74	(00267)	(00268)	(01531)
09C75	(00268)	(00269)	(01531)
09C76	(00269)	(00270)	(01531)
09C77	(00270)	(00271)	(01531)
09C78	(00271)	(00272)	(01531)
09C79	(00272)	(00273)	(01531)
09C80	(00273)	(00274)	(01531)
09C81	(00274)	(00275)	(01531)
09C82	(00275)	(00276)	(01531)
09C83	(00276)	(00277)	(01531)
09C84	(00277)	(00278)	(01531)
09C85	(00278)	(00279)	(01531)
09C86	(00279)	(00280)	(01531)
09C87	(00280)	(00281)	(01531)
09C88	(00281)	(00282)	(01531)
09C89	(00282)	(00283)	(01531)
09C90	(00283)	(00284)	(01531)
09C91	(00284)	(00285)	(01531)
09C92	(00285)	(00286)	(01531)
09C93	(00286)	(00287)	(01531)
09C94	(00287)	(00288)	(01531)
09C95	(00288)	(00289)	(01531)
09C96	(00289)	(00290)	(01531)
09C97	(00290)	(00291)	(01531)
09C98	(00291)	(00292)	(01531)
09C99	(00292)	(00293)	(01531)
09D00	(00293)	(00294)	(01531)
09D01	(00294)	(00295)	(01531)
09D02	(00295)	(00296)	(01531)
09D03	(00296)	(00297)	(01531)
09D04	(00297)	(00298)	(01531)
09D05	(00298)	(00299)	(01531)
09D06	(00299)	(00300)	(01531)
09D07	(00300)	(00301)	(01531)
09D08	(00301)	(00302)	(01531)
09D09	(00302)	(00303)	(01531)
09D			

01C77	(00112)	(00047)		
00553	(00303)	(00705)	(00791)	(00795) (00950) (00977)
0039C	(00220)	(01426)	(01046)	
005A6	(00227)	(01427)	(01901)	
03E0C	(00700)	(00002)	(01427)	(01436)
0053B	(00256)	(00262)	(00739)	
0053C	(00257)	(00740)		
03050	(00740)	(00777)	(00001)	
00547	(00277)	(00776)	(00003)	(00000)
00000	(00233)	(00494)	(00513)	(01053) (01054) (01055)
00EAB	(00234)	(00495)	(00521)	(01060) (01061) (01062)
0057F	(00231)	(00766)		
00547	(00299)	(00004)		
00550	(00300)	(01001)		
02P07	(00342)	(00390)		
03191	(00343)	(00396)		
03390	(00344)	(00402)		
00557	(00157)	(00312)		
0007E	(00223)	(00342)	(00377)	(00404) (00743)
00700	(00224)	(00343)	(00392)	(00744)
00192	(00225)	(00344)	(00390)	(00742) (00745)
0054C	(00200)	(00022)		
030A0	(00709)	(00794)		
030A1	(00793)	(00000)		
030C6	(00770)	(00022)		
0300A	(00775)	(00012)		
030C2	(00762)	(00005)		
030CA	(00760)	(00034)		
03006	(00771)	(00000)		
030C0	(00772)	(00015)		
030AC	(00796)	(00001)		
03062	(00131)	(00150)	(00754)	
00546	(00276)	(00756)	(00760)	
03060	(00745)	(00763)	(00704)	(00797) (00012) (00014)
00570	(00319)	(00323)	(01570)	
00551	(00301)	(00500)	(00765)	
03C00	(01023)	(01035)		
03C04	(01025)	(01036)		
00530	(00250)	(01024)		
0053E	(00259)	(01025)		
00040	(00230)	(01022)		
00C20	(00231)	(01023)		
00C00	(00232)	(01047)		
00540	(00209)	(01046)		
00540	(00279)	(01034)	(01041)	
00700	(00220)	(00070)	(01429)	(01060)
0097C	(00229)	(00079)	(01430)	(01915)
03F10	(01430)	(01439)		
070EC	(00230)	(00074)	(00921)	(00930)
070F6	(00239)	(00009)	(00923)	(00924) (00926) (00937)
00552	(00302)	(00769)	(00774)	(00907) (00909) (00911)
0053F	(00293)	(00295)	(00574)	(00652) (00761) (01032)
00001	(00371)			

PAGE 561

SUMMARY:

PAGE 57: (00007<0C116>00N16U.NSO.2, 29-Dec-88 15:44:23, Ed: WOLF
 APS3-BXC000() APS PROGRAM FOR DEC000

TOPHIST:	0900C (00206)	(01140) (01161)	
TSNAPTR:	03CC4 (01132)	(01131)	
TSNRA:	0A100 (00215)	(01132)	
TSNR0:	0A266 (00216)	(00600)	(00610) (00611)
TSNR1:	09700 (00213)	(00617) (00618)	
TSNR01:	0A0A0 (00214)	(00504)	
TSNAPTR0:	04956 (00564)	(00566)	
TSNRA1:	00534 (00252)	(00260)	
TSNR01:	00535 (00253)	(00567)	
TSNAPTR1:	0495A (00567)	(00577) (00586)	
TSNRA0:	00543 (00271)	(00576) (00580)	
TW0N9:	037CA (01040)	(01070) (01095)	
WS:	00002 (00000)	(00050) (00052)	(00053) (00054) (00055) (00056) (00057) (00058) (00059)
		(00060) (00061) (00062) (00063) (00064) (00065) (00066) (00067) (00068) (00069) (00070)	
		(00071) (00072) (00073) (00074) (00075) (00076) (00077) (00078) (00079) (00080) (00081)	
		(00082) (00083) (00084) (00085) (00086) (00087) (00088) (00089) (00090) (00091) (00092)	
		(00093) (00094) (00095) (00096) (00097) (00098) (00099) (00100) (00101) (00102) (00103)	
		(00104) (00105) (00106) (00107) (00108) (00109) (00110) (00111) (00112) (00113) (00114)	
		(00115) (00116) (00117) (00118) (00119) (00120) (00121) (00122) (00123) (00124) (00125)	
		(00126) (00127) (00128) (00129) (00130) (00131) (00132) (00133) (00134) (00135) (00136)	
		(00137) (00138) (00139) (00140) (00141) (00142) (00143) (00144) (00145) (00146) (00147)	
		(00148) (00149) (00150) (00151) (00152) (00153) (00154) (00155) (00156) (00157) (00158)	
		(00159) (00160) (00161) (00162) (00163) (00164) (00165) (00166) (00167) (00168) (00169)	
		(00170) (00171) (00172) (00173) (00174) (00175) (00176) (00177) (00178) (00179) (00180)	
		(00181) (00182) (00183) (00184) (00185) (00186) (00187) (00188) (00189) (00190) (00191)	
		(00192) (00193) (00194) (00195) (00196) (00197) (00198) (00199) (00200) (00201) (00202)	

LINES WITH ERRORS: 0 (MAP VERSION 00101.10) 2- 0

AD-A096 092

BOLT BERANEK AND NEWMAN INC CAMBRIDGE MA F/G 17/2.1
DESIGN AND REAL-TIME IMPLEMENTATION OF A ROBUST APC CODER FOR S--ETC(U)
DEC 80 J J WOLF, K D FIELD, W H RUSSELL DCA100-79-C-0037
8BN-4565-VOL-2 NL

UNCLASSIFIED

5 OF 5
AQ 6
026092

END
DATE
FILMED
4-81
DTIC

```

C
C
C COMMON MLOCK - BUFFER ADDRESSES AND SIZES
C   NEEDED FOR INIT OF NON-SHAP BUFFERS
C   AND FOR FILE-TO-FILE MODE OF OPERATION
C

```

```

COMMON /M16/
  ) M1STO, M1STO,
  ) M1DBA, M1DBA,
  ) M1DBB, M1DBB,
  ) M1DWC, M1DWC,
  ) M1DTC, M1DTC,
  ) M1DMA, M1DMA,
  ) M1DDB, M1DDB,
  ) M1DWB, M1DWB,
  ) M1DWC, M1DWC,
  ) M1DMA, M1DMA,
  ) M1DDB, M1DDB

```

```

CCCCC END OF MM16-COMMON CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

```

```

C INITIALIZE FORTRAN CONSTANTS:

```

```

DATA PI/3.141592653589793121/
DATA LN2/0.693147180559945309217/
DATA LN10/2.302585092994045684017/
DATA LN5/0.690775527898213659071/

```

```

C DEFINE FUNCTION EVEN(X)
C RETURNS SMALLEST EVEN INTEGER -GE. X
C (ASSUMES X -EQ. AINT(X))

```

```

      EVEN(X) = X + (X - (2.*AINT(X/2.)))

```

```

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

```

```

C CONFIGURE MAP-399 BUFFERS:

```

```

C THIS MODULE CONFIGURES THE SYSTEM MAP BUFFERS.
C ALL MAP BUFFER'S (BIO'S) ARE SYMBOLICALLY
C NAMED, WITH NAMES STARTING WITH "T" FOR
C TRANSMITTER BUFFERS, AND NAMES STARTING WITH
C "R" FOR RECEIVER BUFFERS.
C AN "A" FOLLOWED BY A BUFFER NAME SPECIFIES

```

```

C THE MOST VARIABLE CONTAINING THE ADDRESS OF
C THAT BUFFER. SIMILARLY, AN "S" FOLLOWED BY A
C BUFFER NAME SPECIFIES THE MOST VARIABLE
C CONTAINING THE SIZE OF THAT BUFFER.

```

```

C A #ID OF 0 INDICATES A NON-SHAP BUFFER.

```

```

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

```

```

C DEFINE SYMBOLIC MLOCK'S:

```

```

C DECODED VALUE TABLE OF QUANTIZED PITCH TAPS C1 & C3 (3 BITS)
      DVL1 = 0
      DVL3 = DVL1

```

```

C DECODED VALUE TABLE OF QUANTIZED PITCH TAP C2 (4 BITS)
      DVL2 = 0

```

```

C DECODED VALUE TABLE OF QUANTIZED REFLECTION COEF K1 (6 BITS)
      DVLK1 = 0

```

```

C DECODED VALUE TABLE OF QUANTIZED REFLECTION COEF K2 (5 BITS)
      DVLK2 = 0

```

```

C DECODED VALUE TABLE OF QUANTIZED REFLECTION COEF K3 (4 BITS)
      DVLK3 = 0

```

```

C DECODED VALUE TABLE OF QUANTIZED REFLECTION COEF K4 (4 BITS)
      DVLK4 = 0

```

```

C DECODED VALUE TABLE OF QUANTIZED REFLECTION COEF K5 (4 BITS)
      DVLK5 = 0

```

```

C DECODED VALUE TABLE OF QUANTIZED REFLECTION COEF K6 (4 BITS)
      DVLK6 = 0

```

```

C DECODED VALUE TABLE OF SQR(QUANTIZED ENERGY)
      DVLG = 0

```

```

C DECODED VALUE TABLE OF SQR(QUANTIZED DELTA GAINS)
      DVLDG = 0

```

```

C DECODED VALUE TABLE OF QUANTIZED WESTOVAL SAMPLES
      DVLW = 0

```

```

C DECODED VALUE TABLE OF QUANTIZED WESTOVAL SAMPLES (FOLDED BINARY ORDER)
      DVLWP = 0

```

C <OCAL6>DBM16C.FOR.1 Mon 18-Nov-88 6:32PM Page 114

```

C CODING TABLE OF PITCH TAP THRESHOLDS C1 & C3
  CTHC1 = 0
  CTHC3 = CTHC1

C CODING TABLE OF PITCH TAP THRESHOLDS C2
  CTHC2 = 0

C (CTH1-6 MUST BE PHYSICALLY CONTIGUOUS IN MEMORY)
C CODING TABLE OF A1 THRESHOLDS
  CTHC1 = 0

C CODING TABLE OF A2 THRESHOLDS
  CTHC2 = 0

C CODING TABLE OF A3 THRESHOLDS
  CTHC3 = 0

C CODING TABLE OF A4 THRESHOLDS
  CTHC4 = 0

C CODING TABLE OF A5 THRESHOLDS
  CTHC5 = 0

C CODING TABLE OF A6 THRESHOLDS
  CTHC6 = 0

C CODING TABLE OF ENERGY THRESHOLDS
  CTHC = 0

C CODING TABLE OF DELTA GAIN THRESHOLDS
  CTHDC = 0

C CODING TABLE OF RESIDUAL SAMPLE THRESHOLDS (FOLDED BINARY)
  CTHN = 0

C HISTOGRAM BUFFER FOR A'S (MEMORICING AND SYSTEM MEASUREMENT USE)
  HISTO = 0

C A/D INPUT FROM ADAM
  TADMA = 0

C A/D INPUT FROM ADAM
  TADAB = 0

C A/D 'ON-NOW' TONE DATA
  TADBC = 0

C INPUT SPEECH TO ANL2A
  TSP1 = 1

C INPUT SPEECH TO ANL2B
  TSPN = 2

```

C <OCAL6>DBM16C.FOR.1 Mon 18-Nov-88 6:32PM Page 115

```

C UNITY VECTOR
  TONES = 3

C PREEMPHASIZED SPEECH (CURRENT FRAME)
  TSPN = 4

C PREEMPHASIZED SPEECH (LAST FRAME AND CURRENT FRAME)
  TSP1 = 5

C PREEMPHASIZED SPEECH (LAST 49 OF LAST FRAME, ALL OF CURRENT FRAME)
  TSP = 6

C HANNING WINDOW COEFFICIENTS FOR 265 SAMPLE WINDOW
  THANN = 7

C WINDOWED SPEECH
  TESP = 8

C WINDOWED SPEECH WITH APPENDED ZERUS
  TSPZ = 9

C COSINE TABLE FOR FFT AND IFFT
  TCOST = 10

C FREQUENCY DOMAIN REPRESENTATION OF TSPZ
  C (THIS BUFFER CONTAINS 256 COMPLEX POINTS) A 257TH COMPLEX
  C POINT WILL BE ASSUMED PRESENT DIRECTLY FOLLOWING IT)
  TESP = 11

C REAL PART OF TESP, INCLUDING 257TH POINT
  TESP = 12

C IMAGINARY PART OF TESP, INCLUDING 257TH POINT
  TSP1 = 13

C COMPLEX BUFFER OF TSPSP, TSPSP1
  C (THIS BUFFER CONTAINS 256 COMPLEX POINTS) A 257TH COMPLEX
  C POINT WILL BE ASSUMED PRESENT DIRECTLY FOLLOWING IT)
  TSP = 14

C POWER SPECTRUM OF TESP, COMPRISING REAL PART OF TSPSP,
  C INCLUDING 257TH POINT.
  TSPSP = 15

C ZERUS, COMPRISING IMAGINARY PART OF TSPSP, INCLUDING 257TH POINT
  TSP1 = 16

C AUTOCORRELATION COEFFS FOR PITCH DETERMINATION
  THP = 17

C

```

```

C <CAL6>BMM16C.FUN.1 Mon 18-Nov-88 6:32PM Page 1:6
C <CAL6>BMM16C.FUN.1 Mon 18-Nov-88 6:32PM Page 1:7

C PITCH PREDICTOR COEFFS (TIPS)
TC = 18
C AUTOCORRELATION COEFFS: -TEP(PITCH+M); M=-1,0,+1
TRPP = 19
C SUM AKA FOR MEL
THORE = 28
C CODED PITCH PREDICTOR COEFFS
TIC = 21
C UNANTIZED PITCH PREDICTOR COEFFS
TCM = 22
C FIRST RESIDUAL (CURRENT FRAME)
TE10 = 23
C FIRST RESIDUAL (LAST 6 SAMPLES OF CURRENT FRAME)
TE11 = 24
C FIRST RESIDUAL (LAST 6 SAMPLES OF LAST FRAME, PLUS CURRENT FRAME)
TE1 = 25
C NAMING WINDOW COEFFS FOR 216 SAMPLE WINDOW
THAN1 = 26
C WINDOWED FIRST RESIDUAL
TE1 = 27
C AUTOCORRELATION COEFFS FOR SPECTRAL ANALYSIS
TRS = 28
C LAMBDA = W(M) FOR APC MODULE
TLAU = 29
C MPC'S AUTOCORRELATION COEFFS
TRSP = 30
C REFLECTION COEFFS & ERROR TERMS FROM MELQ6
TRE = 31
C QUANTIZED REFLECTION COEFFS
TKH = 32
C CODED REFLECTION COEFFS
TIA = 33
C PREDICTOR COEFFS A(1)-A(6) (ALSO START OF FILTER COEFF BLOCK)
TAM = 34
TILT = TAN

C <CAL6>BMM16C.FUN.1 Mon 18-Nov-88 6:32PM Page 1:7
C PREDICTION COEFFS (REVERSE BUFFER), WITH A(0)=1 ADDED
TAHR = 35
C SECOND RESIDUAL
TE2 = 36
C FACTORS FOR NOISE SHAPING ANALYSIS
TFAC = 37
C NOISE SHAPING FILTER COEFFS
TANS = 38
C QUANTIZER SCALE FACTORS, ALTERNATING WITH THEIR RECIPROCALLS
TFAC = 39
C CODED DELTA GAINS
TIDG = 40
C UNQUANTIZED, NORMALIZED RESIDUAL (FOR DEBUGGING ONLY)
TUO = 41
C HISTORY FOR NOISE-SHAPING FILTER
TQ1 = 42
C HISTORY FOR SPECTRAL FILTER
TVN = 43
C APC PITCH FILTER HISTORY (CURRENT FRAME)
TRH = 44
C APC PITCH FILTER HISTORY (LAST FRAME PLUS CURRENT FRAME)
TRM = 45
C CODED RESIDUAL SAMPLES FROM ANLZA
C (ALSO, START OF CODED TRANSMISSION DATA BUFFER FROM ANLZA,
C WITH FORMAT T10A,T1C,T1DG,T1M,T1C,T1K)
T10A = 46
T10B = 47
T10C = 48
T10D = 49
T10E = 50
T10F = 51
T10G = 52
T10H = 53
T10I = 54
T10J = 55
T10K = 56
T10L = 57
T10M = 58
T10N = 59
T10O = 60
T10P = 61
T10Q = 62
T10R = 63
T10S = 64
T10T = 65
T10U = 66
T10V = 67
T10W = 68
T10X = 69
T10Y = 70
T10Z = 71
T11A = 72
T11B = 73
T11C = 74
T11D = 75
T11E = 76
T11F = 77
T11G = 78
T11H = 79
T11I = 80
T11J = 81
T11K = 82
T11L = 83
T11M = 84
T11N = 85
T11O = 86
T11P = 87
T11Q = 88
T11R = 89
T11S = 90
T11T = 91
T11U = 92
T11V = 93
T11W = 94
T11X = 95
T11Y = 96
T11Z = 97
T12A = 98
T12B = 99
T12C = 100
T12D = 101
T12E = 102
T12F = 103
T12G = 104
T12H = 105
T12I = 106
T12J = 107
T12K = 108
T12L = 109
T12M = 110
T12N = 111
T12O = 112
T12P = 113
T12Q = 114
T12R = 115
T12S = 116
T12T = 117
T12U = 118
T12V = 119
T12W = 120
T12X = 121
T12Y = 122
T12Z = 123
T13A = 124
T13B = 125
T13C = 126
T13D = 127
T13E = 128
T13F = 129
T13G = 130
T13H = 131
T13I = 132
T13J = 133
T13K = 134
T13L = 135
T13M = 136
T13N = 137
T13O = 138
T13P = 139
T13Q = 140
T13R = 141
T13S = 142
T13T = 143
T13U = 144
T13V = 145
T13W = 146
T13X = 147
T13Y = 148
T13Z = 149
T14A = 150
T14B = 151
T14C = 152
T14D = 153
T14E = 154
T14F = 155
T14G = 156
T14H = 157
T14I = 158
T14J = 159
T14K = 160
T14L = 161
T14M = 162
T14N = 163
T14O = 164
T14P = 165
T14Q = 166
T14R = 167
T14S = 168
T14T = 169
T14U = 170
T14V = 171
T14W = 172
T14X = 173
T14Y = 174
T14Z = 175
T15A = 176
T15B = 177
T15C = 178
T15D = 179
T15E = 180
T15F = 181
T15G = 182
T15H = 183
T15I = 184
T15J = 185
T15K = 186
T15L = 187
T15M = 188
T15N = 189
T15O = 190
T15P = 191
T15Q = 192
T15R = 193
T15S = 194
T15T = 195
T15U = 196
T15V = 197
T15W = 198
T15X = 199
T15Y = 200
T15Z = 201
T16A = 202
T16B = 203
T16C = 204
T16D = 205
T16E = 206
T16F = 207
T16G = 208
T16H = 209
T16I = 210
T16J = 211
T16K = 212
T16L = 213
T16M = 214
T16N = 215
T16O = 216
T16P = 217
T16Q = 218
T16R = 219
T16S = 220
T16T = 221
T16U = 222
T16V = 223
T16W = 224
T16X = 225
T16Y = 226
T16Z = 227
T17A = 228
T17B = 229
T17C = 230
T17D = 231
T17E = 232
T17F = 233
T17G = 234
T17H = 235
T17I = 236
T17J = 237
T17K = 238
T17L = 239
T17M = 240
T17N = 241
T17O = 242
T17P = 243
T17Q = 244
T17R = 245
T17S = 246
T17T = 247
T17U = 248
T17V = 249
T17W = 250
T17X = 251
T17Y = 252
T17Z = 253
T18A = 254
T18B = 255
T18C = 256
T18D = 257
T18E = 258
T18F = 259
T18G = 260
T18H = 261
T18I = 262
T18J = 263
T18K = 264
T18L = 265
T18M = 266
T18N = 267
T18O = 268
T18P = 269
T18Q = 270
T18R = 271
T18S = 272
T18T = 273
T18U = 274
T18V = 275
T18W = 276
T18X = 277
T18Y = 278
T18Z = 279
T19A = 280
T19B = 281
T19C = 282
T19D = 283
T19E = 284
T19F = 285
T19G = 286
T19H = 287
T19I = 288
T19J = 289
T19K = 290
T19L = 291
T19M = 292
T19N = 293
T19O = 294
T19P = 295
T19Q = 296
T19R = 297
T19S = 298
T19T = 299
T19U = 300
T19V = 301
T19W = 302
T19X = 303
T19Y = 304
T19Z = 305
T20A = 306
T20B = 307
T20C = 308
T20D = 309
T20E = 310
T20F = 311
T20G = 312
T20H = 313
T20I = 314
T20J = 315
T20K = 316
T20L = 317
T20M = 318
T20N = 319
T20O = 320
T20P = 321
T20Q = 322
T20R = 323
T20S = 324
T20T = 325
T20U = 326
T20V = 327
T20W = 328
T20X = 329
T20Y = 330
T20Z = 331
T21A = 332
T21B = 333
T21C = 334
T21D = 335
T21E = 336
T21F = 337
T21G = 338
T21H = 339
T21I = 340
T21J = 341
T21K = 342
T21L = 343
T21M = 344
T21N = 345
T21O = 346
T21P = 347
T21Q = 348
T21R = 349
T21S = 350
T21T = 351
T21U = 352
T21V = 353
T21W = 354
T21X = 355
T21Y = 356
T21Z = 357
T22A = 358
T22B = 359
T22C = 360
T22D = 361
T22E = 362
T22F = 363
T22G = 364
T22H = 365
T22I = 366
T22J = 367
T22K = 368
T22L = 369
T22M = 370
T22N = 371
T22O = 372
T22P = 373
T22Q = 374
T22R = 375
T22S = 376
T22T = 377
T22U = 378
T22V = 379
T22W = 380
T22X = 381
T22Y = 382
T22Z = 383
T23A = 384
T23B = 385
T23C = 386
T23D = 387
T23E = 388
T23F = 389
T23G = 390
T23H = 391
T23I = 392
T23J = 393
T23K = 394
T23L = 395
T23M = 396
T23N = 397
T23O = 398
T23P = 399
T23Q = 400
T23R = 401
T23S = 402
T23T = 403
T23U = 404
T23V = 405
T23W = 406
T23X = 407
T23Y = 408
T23Z = 409
T24A = 410
T24B = 411
T24C = 412
T24D = 413
T24E = 414
T24F = 415
T24G = 416
T24H = 417
T24I = 418
T24J = 419
T24K = 420
T24L = 421
T24M = 422
T24N = 423
T24O = 424
T24P = 425
T24Q = 426
T24R = 427
T24S = 428
T24T = 429
T24U = 430
T24V = 431
T24W = 432
T24X = 433
T24Y = 434
T24Z = 435
T25A = 436
T25B = 437
T25C = 438
T25D = 439
T25E = 440
T25F = 441
T25G = 442
T25H = 443
T25I = 444
T25J = 445
T25K = 446
T25L = 447
T25M = 448
T25N = 449
T25O = 450
T25P = 451
T25Q = 452
T25R = 453
T25S = 454
T25T = 455
T25U = 456
T25V = 457
T25W = 458
T25X = 459
T25Y = 460
T25Z = 461
T26A = 462
T26B = 463
T26C = 464
T26D = 465
T26E = 466
T26F = 467
T26G = 468
T26H = 469
T26I = 470
T26J = 471
T26K = 472
T26L = 473
T26M = 474
T26N = 475
T26O = 476
T26P = 477
T26Q = 478
T26R = 479
T26S = 480
T26T = 481
T26U = 482
T26V = 483
T26W = 484
T26X = 485
T26Y = 486
T26Z = 487
T27A = 488
T27B = 489
T27C = 490
T27D = 491
T27E = 492
T27F = 493
T27G = 494
T27H = 495
T27I = 496
T27J = 497
T27K = 498
T27L = 499
T27M = 500
T27N = 501
T27O = 502
T27P = 503
T27Q = 504
T27R = 505
T27S = 506
T27T = 507
T27U = 508
T27V = 509
T27W = 510
T27X = 511
T27Y = 512
T27Z = 513
T28A = 514
T28B = 515
T28C = 516
T28D = 517
T28E = 518
T28F = 519
T28G = 520
T28H = 521
T28I = 522
T28J = 523
T28K = 524
T28L = 525
T28M = 526
T28N = 527
T28O = 528
T28P = 529
T28Q = 530
T28R = 531
T28S = 532
T28T = 533
T28U = 534
T28V = 535
T28W = 536
T28X = 537
T28Y = 538
T28Z = 539
T29A = 540
T29B = 541
T29C = 542
T29D = 543
T29E = 544
T29F = 545
T29G = 546
T29H = 547
T29I = 548
T29J = 549
T29K = 550
T29L = 551
T29M = 552
T29N = 553
T29O = 554
T29P = 555
T29Q = 556
T29R = 557
T29S = 558
T29T = 559
T29U = 560
T29V = 561
T29W = 562
T29X = 563
T29Y = 564
T29Z = 565
T30A = 566
T30B = 567
T30C = 568
T30D = 569
T30E = 570
T30F = 571
T30G = 572
T30H = 573
T30I = 574
T30J = 575
T30K = 576
T30L = 577
T30M = 578
T30N = 579
T30O = 580
T30P = 581
T30Q = 582
T30R = 583
T30S = 584
T30T = 585
T30U = 586
T30V = 587
T30W = 588
T30X = 589
T30Y = 590
T30Z = 591
T31A = 592
T31B = 593
T31C = 594
T31D = 595
T31E = 596
T31F = 597
T31G = 598
T31H = 599
T31I = 600
T31J = 601
T31K = 602
T31L = 603
T31M = 604
T31N = 605
T31O = 606
T31P = 607
T31Q = 608
T31R = 609
T31S = 610
T31T = 611
T31U = 612
T31V = 613
T31W = 614
T31X = 615
T31Y = 616
T31Z = 617
T32A = 618
T32B = 619
T32C = 620
T32D = 621
T32E = 622
T32F = 623
T32G = 624
T32H = 625
T32I = 626
T32J = 627
T32K = 628
T32L = 629
T32M = 630
T32N = 631
T32O = 632
T32P = 633
T32Q = 634
T32R = 635
T32S = 636
T32T = 637
T32U = 638
T32V = 639
T32W = 640
T32X = 641
T32Y = 642
T32Z = 643
T33A = 644
T33B = 645
T33C = 646
T33D = 647
T33E = 648
T33F = 649
T33G = 650
T33H = 651
T33I = 652
T33J = 653
T33K = 654
T33L = 655
T33M = 656
T33N = 657
T33O = 658
T33P = 659
T33Q = 660
T33R = 661
T33S = 662
T33T = 663
T33U = 664
T33V = 665
T33W = 666
T33X = 667
T33Y = 668
T33Z = 669
T34A = 670
T34B = 671
T34C = 672
T34D = 673
T34E = 674
T34F = 675
T34G = 676
T34H = 677
T34I = 678
T34J = 679
T34K = 680
T34L = 681
T34M = 682
T34N = 683
T34O = 684
T34P = 685
T34Q = 686
T34R = 687
T34S = 688
T34T = 689
T34U = 690
T34V = 691
T34W = 692
T34X = 693
T34Y = 694
T34Z = 695
T35A = 696
T35B = 697
T35C = 698
T35D = 699
T35E = 700
T35F = 701
T35G = 702
T35H = 703
T35I = 704
T35J = 705
T35K = 706
T35L = 707
T35M = 708
T35N = 709
T35O = 710
T35P = 711
T35Q = 712
T35R = 713
T35S = 714
T35T = 715
T35U = 716
T35V = 717
T35W = 718
T35X = 719
T35Y = 720
T35Z = 721
T36A = 722
T36B = 723
T36C = 724
T36D = 725
T36E = 726
T36F = 727
T36G = 728
T36H = 729
T36I = 730
T36J = 731
T36K = 732
T36L = 733
T36M = 734
T36N = 735
T36O = 736
T36P = 737
T36Q = 738
T36R = 739
T36S = 740
T36T = 741
T36U = 742
T36V = 743
T36W = 744
T36X = 745
T36Y = 746
T36Z = 747
T37A = 748
T37B = 749
T37C = 750
T37D = 751
T37E = 752
T37F = 753
T37G = 754
T37H = 755
T37I = 756
T37J = 757
T37K = 758
T37L = 759
T37M = 760
T37N = 761
T37O = 762
T37P = 763
T37Q = 764
T37R = 765
T37S = 766
T37T = 767
T37U = 768
T37V = 769
T37W = 770
T37X = 771
T37Y = 772
T37Z = 773
T38A = 774
T38B = 775
T38C = 776
T38D = 777
T38E = 778
T38F = 779
T38G = 780
T38H = 781
T38I = 782
T38J = 783
T38K = 784
T38L = 785
T38M = 786
T38N = 787
T38O = 788
T38P = 789
T38Q = 790
T38R = 791
T38S = 792
T38T = 793
T38U = 794
T38V = 795
T38W = 796
T38X = 797
T38Y = 798
T38Z = 799
T39A = 800
T39B = 801
T39C = 802
T39D = 803
T39E = 804
T39F = 805
T39G = 806
T39H = 807
T39I = 808
T39J = 809
T39K = 810
T39L = 811
T39M = 812
T39N = 813
T39O = 814
T39P = 815
T39Q = 816
T39R = 817
T39S = 818
T39T = 819
T39U = 820
T39V = 821
T39W = 822
T39X = 823
T39Y = 824
T39Z = 825
T40A = 826
T40B = 827
T40C = 828
T40D = 829
T40E = 830
T40F = 831
T40G = 832
T40H = 833
T40I = 834
T40J = 835
T40K = 836
T40L = 837
T40M = 838
T40N = 839
T40O = 840
T40P = 841
T40Q = 842
T40R = 843
T40S = 844
T40T = 845
T40U = 846
T40V = 847
T40W = 848
T40X = 849
T40Y = 850
T40Z = 851
T41A = 852
T41B = 853
T41C = 854
T41D = 855
T41E = 856
T41F = 857
T41G = 858
T41H = 859
T41I = 860
T41J = 861
T41K = 862
T41L = 863
T41M = 864
T41N = 865
T41O = 866
T41P = 867
T41Q = 868
T41R = 869
T41S = 870
T41T = 871
T41U = 872
T41V = 873
T41W = 874
T41X = 875
T41Y = 876
T41Z = 877
T42A = 878
T42B = 879
T42C = 880
T42D = 881
T42E = 882
T42F = 883
T42G = 884
T42H = 885
T42I = 886
T42J = 887
T42K = 888
T42L = 889
T42M = 890
T42N = 891
T42O = 892
T42P = 893
T42Q = 894
T42R = 895
T42S = 896
T42T = 897
T42U = 898
T42V = 899
T42W = 900
T42X = 901
T42Y = 902
T42Z = 903
T43A = 904
T43B = 905
T43C = 906
T43D = 907
T43E = 908
T43F = 909
T43G = 910
T43H = 911
T43I = 912
T43J = 913
T43K = 914
T43L = 915
T43M = 916
T43N = 917
T43O = 918
T43P = 919
T43Q = 920
T43R = 921
T43S = 922
T43T = 923
T43U = 924
T43V = 925
T43W = 926
T43X = 927
T43Y = 928
T43Z = 929
T44A = 930
T44B = 931
T44C = 932
T44D = 933
T44E = 934
T44F = 935
T44G = 936
T44H = 937
T44I = 938
T44J = 939
T44K = 940
T44L = 941
T44M = 942
T44N = 943
T44O = 944
T44P = 945
T44Q = 946
T44R = 947
T44S = 948
T44T = 949
T44U = 950
T44V = 951
T44W = 952
T44X = 953
T44Y = 954
T44Z = 955
T45A = 956
T45B = 957
T45C = 958
T45D = 959
T45E = 960
T45F = 961
T45G = 962
T45H = 963
T45I = 964
T45J = 965
T45K = 966
T45L = 967
T45M = 968
T45N = 969
T45O = 970
T45P = 971
T45Q = 972
T45R = 973
T45S = 974
T45T = 975
T45U = 976
T45V = 977
T45W = 978
T45X = 979
T45Y = 980
T45Z = 981
T46A = 982
T46B = 983
T46C = 984
T46D = 985
T46E = 986
T46F = 987
T46G = 988
T46H = 989
T46I = 990
T46J = 991
T46K = 992
T46L = 993
T46M = 994
T46N = 995
T46O = 996
T46P = 997
T46Q = 998
T46R = 999
T46S = 1000

```


C <SCALE>BMB10C.FDM.1 Mon 18-Nov-88 6:32PM

C
C
C TSMA BUFFER STATUS FLAG
TSMPA = 50

C
C TSMB BUFFER STATUS FLAG
TSMPB = 51

C
C TSMA BUFFER STATUS FLAG
TSMPA = 52

C
C TSMB BUFFER STATUS FLAG
TSMPB = 53

C
C NMLP OUTPUT: NUMBER OF ITERATIONS NOT COMPLETED (TINC,TENC COUNTG.)
TINC = 54

C
C NMLP OUTPUT: FRACTIONAL ENERGY E(N)/R(0) (TINC,TENC COUNTG.)
TENC = 55

C (UNUSE)
1 = 56

C
C DATA BUFFER STATUS FLAG
MSDPA = 57

C
C DATA BUFFER STATUS FLAG
MSDPB = 58

C
C BSMKA BUFFER STATUS FLAG
BSMPA = 59

C
C BSMKB BUFFER STATUS FLAG
BSMPB = 60

C
C SYSTEM RUN FLAG (0 -> STOP)
MUN = 61

C
C (LEFT UNUSED FOR MP1TH)
1 = 62

C (UNUSE)
1 = 63

C
C TSMA/B BUFFER POINTER (0->B) -2->A)
TSMPA = 64

C
C TSMA/B BUFFER POINTER (0->B) -2->A)
TSMPB = 65

L <SCALE>BMB10C.FDM.1 Mon 18-Nov-88 6:32PM Page 1:1B

C
C TSMA/B BUFFER POINTER (0->B) -2->A)
TSMPA = 66

C
C TSMA/B BUFFER POINTER (0->B) -2->A)
TSMPB = 67

C
C TSMA/B/C BUFFER POINTER (0->C) -2->B) -4->A)
TSMPD = 68

C
C TSMA/B BUFFER POINTER (0->B) -2->A)
TSMPB = 69

C
C BSMKA/B BUFFER POINTER (0->B) -2->A)
BSMPA = 70

C
C BSMKB/B BUFFER POINTER (0->B) -2->A)
BSMPB = 71

C
C A/D FRAME DISCARD COUNT
TAUDC = 72

C
C THODEM FIVE FRAME COUNTER
THFEC = 73

C
C THODEM FRAME DISCARD COUNTER
THFDC = 74

C
C BSINK DATA-NOT-READY COUNTER
BSNDR = 75

C
C TRANSMITTER FRAME COUNTER
TFCTR = 76

C
C RECEIVER FRAME COUNTER
RFCTR = 77

C
C RECEIVER LUST-SYNC COUNTER
RLCTR = 78

C
C LOCAL HANDSET ON-HOOK STATE (0->HANDSET OFF-HOOK)
KONHR = 79

C
C STATE OF RECEIVER SYNC (0->LUST SYNC, -1->SEARCHING SYNC) 1->HAVE SYNC.
RSYNC = 80

C
C BEGINNING-OF-FRAME OFFSET (SYNC BIT POSITION IN THODEM BUFFER)
MBOFU = 81

C
C (LEFT UNUSED FOR MP1TH)
1 = 82

C <LOCAL>MM16C.FOR.1 Mon 18-Nov-88 6:32PM

```

C A/D (ADM) INTERRUPT COUNTER
  ADCIN = 83
C TIMER (IOS-2) INTERRUPT COUNTER
  TMCIN = 84
C ANOUEM (IOS-2) INTERRUPT COUNTER
  ANOUEM = 85
C D/A (ADM) INTERRUPT COUNTER
  DACIN = 86
C A/D PEAK INPUT LEVEL
  TADPE = 87
C NO-ERROR-CORRECTION SWITCH (NO CORRECTIONS IF M2)
  INCOR = 123
C (LEFT UNUSED FOR MP1M)
  I = 124
C CHANNEL ERROR SIMULATOR FLAG (NON-ZERO => 8 SIM ERRORS PER FRAME)
  ERRS = 125
C (LEFT BRUSED FOR MP1M)
  I = 126
C
C
C
C
C
  RETURN
  END

```



```
C
      INTEGER FUNCTION KCLUSK(I,MMAY)
      DIMENSION IARRAY(1)
      TYPE 100
      100 FORMAT(' THIS MODE OF OPERATION NOT YET FULLY IMPLEMENTED.')
      STOP
      END

C
C
C      SUBROUTINE DELAY(I,J)
      TYPE 100
      100 FORMAT(' THIS MODE OF OPERATION NOT YET FULLY IMPLEMENTED.')
      STOP
      END
```

```

C <DCAL16>BHM16E.FOR.1 Mon 18-Nov-80 6:35PM
C
C MAP BUFFER NAMES:
C
INTEGER DVLCL,DVLC2,DVLC3,DVLK1,DVLK2,DVLK3
INTEGER DVLK4,DVLK5,DVLK6,DVLG,DVLG,DVLG
INTEGER DVLUV
INTEGER CTHK1,CTHC2,CTHC3,CTHK1,CTHK2,CTHK3
INTEGER CTHK4,CTHK5,CTHK6,CTHG,CTHKG,CTHU
INTEGER DISTS
INTEGER TAUBA,TAUBM,TAUBC,TSMA,TSMB,TUNE3
INTEGER TSPB,TSP1,TSP,THAB,THSP,TSP2
INTEGER TCST,TSP,TSPM,TASPI,TSP,TSPM
INTEGER TSP1,TSP,TC,TWMA,TIC,TCA
INTEGER T218,T211,T21,THAM,TM1,TM3
INTEGER TLMU,TREP,TIE,TKH,TIK,TFLT
INTEGER TAB,TABR,T2,TAC,TAS,TGAC
INTEGER TIUG,TU,TU,TU,TU,TU,TU,TU
INTEGER TIUA,TIUB,TSMA,TSMB,TSTA,TSTB
INTEGER TSTC,TDMA,TDMB,TDP
INTEGER HMDA,HUMB,HMHC,HSTA,HSTB,MSRA
INTEGER ASMB,MUMB,MHCH,MCA,MCA,MCA
INTEGER RVH,RVH,RVH,RVH,RVH,RVH,RVH
INTEGER RRND,RSMA,RSMB,RSNC,RDAB,RDAB
C
C MAP REAL SCALAR NAMES:
C
INTEGER TALP1,TPEST,TDCH,TFSZ1,TNH,TIM
INTEGER TI4,THAM,TIC,IC2P,T2R,T2M
INTEGER RCH,RH,RDCF,RDCF,RDCF,RDCF
INTEGER RDMY,RDMY1,RDMY2,RDMY3
C
C MAP INTEGER SCALAR NAMES:
C
INTEGER TSMA,TSMB,TSTA,TSTB,TSTC,TSTC
INTEGER RSTB,RSTB,RSMA,RSMB,RUN
INTEGER ADPO,TSPO,TSTO,TPO,RMPO,RTPO
INTEGER ASMP,DAPO,TADDC,TMFC,RMFC,MSMR
INTEGER TFCM,HCTH,MCTA,RMNH,MSMC,MDFU
INTEGER ADCR,THCT,RMCT,DACT,TAUP
INTEGER RNCOR,REARS
C
C COMMON BLOCK - BUFFER AND SCALAR NAMES:
C
COMMON /BIDS/
) DVLCL,DVLC2,DVLC3,DVLK1,DVLK2,DVLK3,
) DVLK4,DVLK5,DVLK6,DVLG,DVLG,DVLG,
) DVLUV,
) CTHK1,CTHC2,CTHC3,CTHK1,CTHK2,CTHK3,
) CTHK4,CTHK5,CTHK6,CTHG,CTHKG,CTHU,

```

```

C <DCAL16>BHM16E.FOR.1 Mon 18-Nov-80 6:35PM Page 1
C
C (BHM16)DCAL16>BHM16E.FOR.1, 18-Nov-80 18:35:10, RUI AF1111
SUBROUTINE BHM16
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C BHM16 SPEECH CODER SYSTEM EXECUTION MODULE
C
C
C EXECUTION MODE ENTERED BY USER:
C 1) REAL TIME SYSTEM
C 2) FILE TO FILE SYSTEM
C 3) TIMING SYSTEM
C 4) SCPE TIMING SYSTEM
C
C CALLED BY BHM16R
C
C RFIELD 9/80
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C MOST ARRAYS FOR FILE TO FILE TEST SYSTEM
C
INTEGER XSIM(264),XSUT(264)
INTEGER HOBLE(256),MSG(20)
INTEGER SIG(238),IST(87)
REAL SIG(238)
EQUIVALENCE (HOBLE(1),MSG(1),SIG(1),IST(1))
LOGICAL CPMH,LTSME,LIRST,LOOPS,BBOFS
C
C MAP FUNCTION LIST NAMES
C
INTEGER BBRYS,BBMLP,BBFFT,BBNTIM,BBHRST
C
OTHER MAP-RELATED VARIABLES
C
REAL HWSZ,FWSZ,BWSZ
INTEGER RL,CPLI,PXD,CNTG,LNG,SHRT
INTEGER PRCSR,AP,NE,MIS
C
LOGICAL *1 CHRC,CHRE,CHRL,CHRR,CHRS,CHRY,CHRQ,CHRON,ICND
REAL HOMPBC,ADMPBC,ADMPBC
INTEGER HOMSCL,ADAM,ADM,IOS2
C
C
C
C (BHM16)MAP>BHM16-COMMON.FOR.12, 31-Oct-80 23:17:03, RUI: *OLF
C

```


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

C EXECUTE FILE TO FILE SYSTEM

C CONTINUE

C

C DEFINE INPUT AND OUTPUT FILES

C

C TYPE 210

C FORMAT(' READ INPUT SIGNAL FROM FILE: '\$)

C NBLKS = KOPR(KSIH,M)

C IF(NBLKS.LE.0) GO TO 203

C CALL SCOPY('SKIP HEADER BLOCK (256 WORDS)? ',MSG)

C IF (.NOT. CMPIRM(MSG)) GO TO 215

C ISTAT = KSIH(MDABLE,256,MDFRD)

C TYPE 220

C FORMAT(' WRITE OUTPUT SIGNAL TO FILE: '\$)

C NBLKS = KOPR(KSOT,0,0,TRUE.)

C IF(NBLKS.LE.0) GO TO 215

C USE DUMMY BUFFER TO READ & WRITE TADDA,M & MDAJA,M

C SINCE THEY ARE RL,SHORT FORMAT, AND MPDOR/MPROB

C DON'T PROPERLY ACCESS RL,SHORT BUFFERS.

C USE BUFFER FROM SYSTEM WHICH IS CORRECT LENGTH (210

C SAMPLES), RL, LMG, AND IS NOT NEEDED FOR FRAME-TU-

C FRAME MEMORY.

C IDUM = 1481

C IADBUF = 60

C IADBUF = 61

C IADUM = 60

C IADMA = 61

C INITIALIZE SYNC BIT OF RMDMA TO 1 FOR FIRST HINT CALL

C (BY USING INITIALIZED TADMB BUFFER), SO 2ND HINT CALL

C (WITH VALID DATA IN RMDMB) SYNC BIT = 0) WILL SHOW

C ALTERNATING SYNC BIT.

C

C "Q" COMMAND -- QUIT (HALT SPEECH CODE)

C CALL MP1TH(RUN,0,0)

C CALL MP1TH(RUN,0,0)

C (THERE'S NO WAY TO STOP THE MUDEN SCROLL)

C RETURN TO COMMON CUMS

C GO TO 900

C "E" COMMAND -- SET ERROR SIMULATION

C IF(B.LT.0 .OR. B.GT.50) GO TO 115

C CALL MP1TH(RERRS,0,0)

C GO TO 112

C "C" COMMAND -- CONTROL ERROR CORRECTION

C IF(B.NE.0) B=1

C CALL MP1TH(RCOR,0,0)

C GO TO 112

C "L" COMMAND -- CAUSE THE PCVR TO LOSE SYNC BY JAMMING INTO RMDPU

C CALL MP1TH(RMDPU,0,0)

C GO TO 112

C "T" COMMAND -- STOP CODE, READ INTEGER SCALARS, RESET TADPA

C (PEAK INPUT LEVEL SO FAR) TO ZERO, THEN RESTART,

C AND TYPE OUT INTEGER SCALAR VALUES

C CALL MP1TH(RDM,0,0)

C CALL MP1TH(TSRPA,1ST(TSRPA),0,0)

C CALL MP1TH(TADPE,0)

C CALL MP1TH(RDMNST)

C CALL TST(1ST)

C GO TO 112

C "S" COMMAND -- SUSPEND THIS TASK, BUT LET MAP CONTINUE.

C (BPOB RESUMPTION OF TASK, IT WILL CONTINUE OUT OF THE PAUSE)

C PAUSE '(SPEECH CODE CONTINUING)'

C GO TO 110

C

Line	Code	Text	Page	Date	Time
210	C	ASK ABOUT FRAME TYPES	11	10-Nov-88	6:35PM
211	C	LIST--FALSE.	11	10-Nov-88	6:35PM
212	C	TYPE 216	11	10-Nov-88	6:35PM
213	C	FORMAT("FRAME TYPEOUTS? (TYPE, "8-CALARS, OR MIL): ")	11	10-Nov-88	6:35PM
214	C	ACCEPT 227,ICMD	11	10-Nov-88	6:35PM
215	C	FORMAT(1)	11	10-Nov-88	6:35PM
216	C	IF(ICMD.EQ.CHRS) LIST--TRUE.	11	10-Nov-88	6:35PM
217	C	IF(ICMD.NE.CHRS) GO TO 230	11	10-Nov-88	6:35PM
218	C	LIST--TRUE.	11	10-Nov-88	6:35PM
219	C	TYPE 228	11	10-Nov-88	6:35PM
220	C	FORMAT("FILE FOR TSINK DATA: ")	11	10-Nov-88	6:35PM
221	C	CALL ASSIGN(2,ISIG,-1)	11	10-Nov-88	6:35PM
222	C	WRITE(2,229)	11	10-Nov-88	6:35PM
223	C	FORMAT(" " F40 6 D11 D12 D13 4 C1 C2 C3 K1 "	11	10-Nov-88	6:35PM
224	C	,"E2 K3 K4 K5 K6 RESIGNAL")	11	10-Nov-88	6:35PM
225	C	TYPE OF LOOP	11	10-Nov-88	6:35PM
226	C	1STAT = KSIG(ISIG,ISIG,216,MWIFRO)	11	10-Nov-88	6:35PM
227	C	IF (1STAT.EQ. IEOP) GO TO 230	11	10-Nov-88	6:35PM
228	C	DO 235 I=1,216	11	10-Nov-88	6:35PM
229	C	SIG(I) = FLOOR(ISIG(I) / 32768.	11	10-Nov-88	6:35PM
230	C	SAT FLAG FOR "A" OR "M" MWIFERS	11	10-Nov-88	6:35PM
231	C	ABUFS = .FALSE.	11	10-Nov-88	6:35PM
232	C	IF(MOD(MWIFERS,2).EQ.0) ABUFS = .TRUE.	11	10-Nov-88	6:35PM
233	C	ABUFS = .NOT. ABUFS	11	10-Nov-88	6:35PM
234	C	IF (ABUFS) ITSINK=TSINKA	11	10-Nov-88	6:35PM
235	C	IF (ABUFS) ITSINK=TSINKB	11	10-Nov-88	6:35PM
236	C	IFAB SIG TO MAP	11	10-Nov-88	6:35PM
237	C	CALL MPWDB(IOUN,SIG,4,1,SIG(216))	11	10-Nov-88	6:35PM
238	C	IF (ABUFS) CALL MPCLB(IUBS1,IABUUF,ATADBA,STADBA,RL,CNTG,SHRT)	11	10-Nov-88	6:35PM
239	C	IF (ABUFS) CALL MPCLB(IUBS1,IABUUF,ATADBB,STADBB,RL,CNTG,SHRT)	11	10-Nov-88	6:35PM
240	C	CALL MPWT(PRCR,MS)	11	10-Nov-88	6:35PM
241	C	CALL VMOV(IABUUF,IOUN)	11	10-Nov-88	6:35PM
242	C	CALL MPWT(PRCR,AP)	11	10-Nov-88	6:35PM
243	C	EXECUTE 1 PASS THRU SYSTEM LOOP	11	10-Nov-88	6:35PM
244	C	CALL MPWFL(OWMPT)	11	10-Nov-88	6:35PM
245	C	MAKE MUST WAIT 12016.67 MSEC = 200 MSEC.	11	10-Nov-88	6:35PM
246	C	CALL DELAY(2,12)	11	10-Nov-88	6:35PM
247	C	END MOVE BITS FROM TRUNK BUFFER TO REMOTE BUFFER, AS A CHANNEL	11	10-Nov-88	6:35PM
248	C	DOES.	11	10-Nov-88	6:35PM
249	C	GO TO(241,242,243,244,245,246) I=MOD(MWIFERS,6)	11	10-Nov-88	6:35PM

```

C <OCAL0>>BUBBLE.FOR.1 Mon 18-Nov-88 6:13PM Page 3:3
C IF REQUESTED, DO FRAME TYPEOUT
IF(.NOT.LTSM) GO TO 260
CALL MPFOR(LTSM,ISIG,2,0,ISIG(230))
WRITE(2,255) NFRAME,(ISIG(I), I=217,230),(ISIG(I),I=1,216)
FORMAT(15,15,1414,/,7211,/,7211,/,7211,/)
255 IF(LIST) CALL LIST
260 NFRAME=NFRAME+1
GO TO 270
C
C HERE WE END.
C
290 CALL LIST
TYPE 2%,NFRAME,MCLIPS
FORMAT(17," FRAMES=",15," OUTPUT SAMPLES WERE CLIPPED.")
LISTAT = KCLOSE(ISIG)
LISTAT = KCLOSE(LSUT)
IF(LTSM) CALL CLOSE(2)
C
C RETURN TO COMMON CODE
C
COTO 900
~L
C <UCAL0>>BUBBLE.FOR.1 Mon 18-Nov-88 6:13PM Page 4
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C EXECUTE TIMING SYSTEM
C
300 CONTINUE
C
C GET TIMING PARAMETERS.
C
TYPE 310
FORMAT(" ENTER ITERATION VALUE: ")
ACCT 311,M
311 FORMAT(I3)
C
C PUT SOME TIMING REASONABLE IN TAUMA, TADMB
C (DON'T USE BID 63 SINCE VCOST USES IT IMPLICITLY.)
C
IADMB = 60
CALL MCLK(TAUMSI,IADMB,ATAUMA,STAUMA,NL,CNTG,SHMT)
CALL VCOST(IADMB,I7.)
CALL MCLK(TADMSI,IADMB,ATADMB,STADMB,NL,CNTG,SHMT)
CALL VCOST(IADMB,I7.)
C
C SET UP TIMING (100 USEC TICKS)
READ CLOCK
C
CALL SETCLK(I)
IT1 = IRDCLK(0)
C
EXECUTE DBTIM "M" TIMES
C
CALL MPST(QUN,I)
CALL MPFOR(I,I,B,DBTIM)
C
WAIT ON MAP, READ CLOCK
C
CALL MPST(I,NONE,I,1)
IT2 = IRDCLK(0)
TIME = FLAT((OCLF(IT1,IT2)))
C
CONVERT TIME, PRINT IT
C
TIMENS = TIME * .1
TYPE 320,M,TIMENS
FORMAT(16," ITERATION(S) TOOK",F10.4," MSEC.S.")
320
C
RETURN TO COMMON CODE
C
COTO 900
~L

```

```

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C      EXECUTE SCODE TUNING SYSTEM
C
C      400 CONTINUE
C
C      PUT SOMETHING REASONABLE IN TADMA,TADDB
C      (DON'T USE DIV 63 SINCE VCOST USES IT IMPLICITLY..)
C
C      TADDB = 60
C      CALL WCLB((IUBS1:TADDBF,ATADMA,STADBA,RL,COTG,SHRT)
C      CALL VCOST(TADDBF,17.)
C      CALL WCLB((IUBS1:TADDBF,ATADDB,STADDB,RL,COTG,SHRT)
C      CALL VCOST(TADDBF,17.)
C
C      START UP INFINITE MPIDL UN BUNTIN
C
C      CALL MPIDL(RUN,1)
C      CALL MPIDL(RUN,86,0,BUNTIN)
C
C      WAIT FOR USER TO SAY QUIT
C
C      TYPE 410
C      FORMAT(' TYPE ANY CHARACTER TO HALT MAP1 ')
C      ACCEPT 411,ICAR
C      FORMAT(A1)
C
C      STOP MPIDL
C
C      CALL MPIDL(RUN,0,0)
C
C      RETURN TO COMMON CUBE
C
C      COTO 906
C
C      END

```

```

SUBROUTINE RIST
C      READ IST AND TYPE OUT CONTENTS
C      DIMENSION IST(87)
C
C      CALL MPIDL(1,IST(1),87,0)
C      CALL TIST(IST)
C
C      RETURN
C      END
C
C      SUBROUTINE TIST(IST)
C      TYPE OUT SELECTED VALUES FROM IST ARRAY
C
C      (UNBND3<MAP>000106-COMMON.F00.12, 31-Oct-80 23:17:03), EJS:WJLP
C
C      C      MAP BUFFER NAMES:
C
C      INTEGER DVLC1,DVLC2,DVLC3,DVLC4,DVLC5,DVLC6,DVLC7,DVLC8,DVLC9
C      INTEGER DVLC4,DVLC5,DVLC6,DVLC7,DVLC8,DVLC9
C      INTEGER DVLCUP
C      INTEGER CTMC1,CTMC2,CTMC3,CTMC4,CTMC5,CTMC6,CTMC7,CTMC8,CTMC9
C      INTEGER HISTO
C      INTEGER TADBA,TADDB,TADDC,TSRA,TSRB,TSRC,TSRD,TSRE,TSRF,TSRG,TSRH,TSRI,TSRJ,TSRK,TSRL,TSRM,TSRN,TSRO,TSRP,TSRQ,TSRR,TSRS,TSRT,TSRU,TSRV,TSRW,TSRX,TSRY,TSRZ
C      INTEGER TCUST,TISP,TRSPM,TRSPN,TRSPQ,TRSPR,TRSPS,TRSPV,TRSPW,TRSPX,TRSPY,TRSPZ
C      INTEGER TSP1,TRP,TC,TWORE,TIC,TCH
C      INTEGER TE10,TE11,TE1,TE2,TE3,TE4,TE5,TE6,TE7,TE8,TE9,TE10,TE11,TE12,TE13,TE14,TE15,TE16,TE17,TE18,TE19,TE20,TE21,TE22,TE23,TE24,TE25,TE26,TE27,TE28,TE29,TE30,TE31,TE32,TE33,TE34,TE35,TE36,TE37,TE38,TE39,TE40,TE41,TE42,TE43,TE44,TE45,TE46,TE47,TE48,TE49,TE50,TE51,TE52,TE53,TE54,TE55,TE56,TE57,TE58,TE59,TE60,TE61,TE62,TE63,TE64,TE65,TE66,TE67,TE68,TE69,TE70,TE71,TE72,TE73,TE74,TE75,TE76,TE77,TE78,TE79,TE80,TE81,TE82,TE83,TE84,TE85,TE86,TE87,TE88,TE89,TE90,TE91,TE92,TE93,TE94,TE95,TE96,TE97,TE98,TE99,TE100,TE101,TE102,TE103,TE104,TE105,TE106,TE107,TE108,TE109,TE110,TE111,TE112,TE113,TE114,TE115,TE116,TE117,TE118,TE119,TE120,TE121,TE122,TE123,TE124,TE125,TE126,TE127,TE128,TE129,TE130,TE131,TE132,TE133,TE134,TE135,TE136,TE137,TE138,TE139,TE140,TE141,TE142,TE143,TE144,TE145,TE146,TE147,TE148,TE149,TE150,TE151,TE152,TE153,TE154,TE155,TE156,TE157,TE158,TE159,TE160,TE161,TE162,TE163,TE164,TE165,TE166,TE167,TE168,TE169,TE170,TE171,TE172,TE173,TE174,TE175,TE176,TE177,TE178,TE179,TE180,TE181,TE182,TE183,TE184,TE185,TE186,TE187,TE188,TE189,TE190,TE191,TE192,TE193,TE194,TE195,TE196,TE197,TE198,TE199,TE200,TE201,TE202,TE203,TE204,TE205,TE206,TE207,TE208,TE209,TE210,TE211,TE212,TE213,TE214,TE215,TE216,TE217,TE218,TE219,TE220,TE221,TE222,TE223,TE224,TE225,TE226,TE227,TE228,TE229,TE230,TE231,TE232,TE233,TE234,TE235,TE236,TE237,TE238,TE239,TE240,TE241,TE242,TE243,TE244,TE245,TE246,TE247,TE248,TE249,TE250,TE251,TE252,TE253,TE254,TE255,TE256,TE257,TE258,TE259,TE260,TE261,TE262,TE263,TE264,TE265,TE266,TE267,TE268,TE269,TE270,TE271,TE272,TE273,TE274,TE275,TE276,TE277,TE278,TE279,TE280,TE281,TE282,TE283,TE284,TE285,TE286,TE287,TE288,TE289,TE290,TE291,TE292,TE293,TE294,TE295,TE296,TE297,TE298,TE299,TE300,TE301,TE302,TE303,TE304,TE305,TE306,TE307,TE308,TE309,TE310,TE311,TE312,TE313,TE314,TE315,TE316,TE317,TE318,TE319,TE320,TE321,TE322,TE323,TE324,TE325,TE326,TE327,TE328,TE329,TE330,TE331,TE332,TE333,TE334,TE335,TE336,TE337,TE338,TE339,TE340,TE341,TE342,TE343,TE344,TE345,TE346,TE347,TE348,TE349,TE350,TE351,TE352,TE353,TE354,TE355,TE356,TE357,TE358,TE359,TE360,TE361,TE362,TE363,TE364,TE365,TE366,TE367,TE368,TE369,TE370,TE371,TE372,TE373,TE374,TE375,TE376,TE377,TE378,TE379,TE380,TE381,TE382,TE383,TE384,TE385,TE386,TE387,TE388,TE389,TE390,TE391,TE392,TE393,TE394,TE395,TE396,TE397,TE398,TE399,TE400,TE401,TE402,TE403,TE404,TE405,TE406,TE407,TE408,TE409,TE410,TE411,TE412,TE413,TE414,TE415,TE416,TE417,TE418,TE419,TE420,TE421,TE422,TE423,TE424,TE425,TE426,TE427,TE428,TE429,TE430,TE431,TE432,TE433,TE434,TE435,TE436,TE437,TE438,TE439,TE440,TE441,TE442,TE443,TE444,TE445,TE446,TE447,TE448,TE449,TE450,TE451,TE452,TE453,TE454,TE455,TE456,TE457,TE458,TE459,TE460,TE461,TE462,TE463,TE464,TE465,TE466,TE467,TE468,TE469,TE470,TE471,TE472,TE473,TE474,TE475,TE476,TE477,TE478,TE479,TE480,TE481,TE482,TE483,TE484,TE485,TE486,TE487,TE488,TE489,TE490,TE491,TE492,TE493,TE494,TE495,TE496,TE497,TE498,TE499,TE500,TE501,TE502,TE503,TE504,TE505,TE506,TE507,TE508,TE509,TE510,TE511,TE512,TE513,TE514,TE515,TE516,TE517,TE518,TE519,TE520,TE521,TE522,TE523,TE524,TE525,TE526,TE527,TE528,TE529,TE530,TE531,TE532,TE533,TE534,TE535,TE536,TE537,TE538,TE539,TE540,TE541,TE542,TE543,TE544,TE545,TE546,TE547,TE548,TE549,TE550,TE551,TE552,TE553,TE554,TE555,TE556,TE557,TE558,TE559,TE560,TE561,TE562,TE563,TE564,TE565,TE566,TE567,TE568,TE569,TE570,TE571,TE572,TE573,TE574,TE575,TE576,TE577,TE578,TE579,TE580,TE581,TE582,TE583,TE584,TE585,TE586,TE587,TE588,TE589,TE590,TE591,TE592,TE593,TE594,TE595,TE596,TE597,TE598,TE599,TE600,TE601,TE602,TE603,TE604,TE605,TE606,TE607,TE608,TE609,TE610,TE611,TE612,TE613,TE614,TE615,TE616,TE617,TE618,TE619,TE620,TE621,TE622,TE623,TE624,TE625,TE626,TE627,TE628,TE629,TE630,TE631,TE632,TE633,TE634,TE635,TE636,TE637,TE638,TE639,TE640,TE641,TE642,TE643,TE644,TE645,TE646,TE647,TE648,TE649,TE650,TE651,TE652,TE653,TE654,TE655,TE656,TE657,TE658,TE659,TE660,TE661,TE662,TE663,TE664,TE665,TE666,TE667,TE668,TE669,TE670,TE671,TE672,TE673,TE674,TE675,TE676,TE677,TE678,TE679,TE680,TE681,TE682,TE683,TE684,TE685,TE686,TE687,TE688,TE689,TE690,TE691,TE692,TE693,TE694,TE695,TE696,TE697,TE698,TE699,TE700,TE701,TE702,TE703,TE704,TE705,TE706,TE707,TE708,TE709,TE710,TE711,TE712,TE713,TE714,TE715,TE716,TE717,TE718,TE719,TE720,TE721,TE722,TE723,TE724,TE725,TE726,TE727,TE728,TE729,TE730,TE731,TE732,TE733,TE734,TE735,TE736,TE737,TE738,TE739,TE740,TE741,TE742,TE743,TE744,TE745,TE746,TE747,TE748,TE749,TE750,TE751,TE752,TE753,TE754,TE755,TE756,TE757,TE758,TE759,TE760,TE761,TE762,TE763,TE764,TE765,TE766,TE767,TE768,TE769,TE770,TE771,TE772,TE773,TE774,TE775,TE776,TE777,TE778,TE779,TE780,TE781,TE782,TE783,TE784,TE785,TE786,TE787,TE788,TE789,TE790,TE791,TE792,TE793,TE794,TE795,TE796,TE797,TE798,TE799,TE800,TE801,TE802,TE803,TE804,TE805,TE806,TE807,TE808,TE809,TE810,TE811,TE812,TE813,TE814,TE815,TE816,TE817,TE818,TE819,TE820,TE821,TE822,TE823,TE824,TE825,TE826,TE827,TE828,TE829,TE830,TE831,TE832,TE833,TE834,TE835,TE836,TE837,TE838,TE839,TE840,TE841,TE842,TE843,TE844,TE845,TE846,TE847,TE848,TE849,TE850,TE851,TE852,TE853,TE854,TE855,TE856,TE857,TE858,TE859,TE860,TE861,TE862,TE863,TE864,TE865,TE866,TE867,TE868,TE869,TE870,TE871,TE872,TE873,TE874,TE875,TE876,TE877,TE878,TE879,TE880,TE881,TE882,TE883,TE884,TE885,TE886,TE887,TE888,TE889,TE890,TE891,TE892,TE893,TE894,TE895,TE896,TE897,TE898,TE899,TE900,TE901,TE902,TE903,TE904,TE905,TE906,TE907,TE908,TE909,TE910,TE911,TE912,TE913,TE914,TE915,TE916,TE917,TE918,TE919,TE920,TE921,TE922,TE923,TE924,TE925,TE926,TE927,TE928,TE929,TE930,TE931,TE932,TE933,TE934,TE935,TE936,TE937,TE938,TE939,TE940,TE941,TE942,TE943,TE944,TE945,TE946,TE947,TE948,TE949,TE950,TE951,TE952,TE953,TE954,TE955,TE956,TE957,TE958,TE959,TE960,TE961,TE962,TE963,TE964,TE965,TE966,TE967,TE968,TE969,TE970,TE971,TE972,TE973,TE974,TE975,TE976,TE977,TE978,TE979,TE980,TE981,TE982,TE983,TE984,TE985,TE986,TE987,TE988,TE989,TE990,TE991,TE992,TE993,TE994,TE995,TE996,TE997,TE998,TE999,TE1000,TE1001,TE1002,TE1003,TE1004,TE1005,TE1006,TE1007,TE1008,TE1009,TE1010,TE1011,TE1012,TE1013,TE1014,TE1015,TE1016,TE1017,TE1018,TE1019,TE1020,TE1021,TE1022,TE1023,TE1024,TE1025,TE1026,TE1027,TE1028,TE1029,TE1030,TE1031,TE1032,TE1033,TE1034,TE1035,TE1036,TE1037,TE1038,TE1039,TE1040,TE1041,TE1042,TE1043,TE1044,TE1045,TE1046,TE1047,TE1048,TE1049,TE1050,TE1051,TE1052,TE1053,TE1054,TE1055,TE1056,TE1057,TE1058,TE1059,TE1060,TE1061,TE1062,TE1063,TE1064,TE1065,TE1066,TE1067,TE1068,TE1069,TE1070,TE1071,TE1072,TE1073,TE1074,TE1075,TE1076,TE1077,TE1078,TE1079,TE1080,TE1081,TE1082,TE1083,TE1084,TE1085,TE1086,TE1087,TE1088,TE1089,TE1090,TE1091,TE1092,TE1093,TE1094,TE1095,TE1096,TE1097,TE1098,TE1099,TE1100,TE1101,TE1102,TE1103,TE1104,TE1105,TE1106,TE1107,TE1108,TE1109,TE1110,TE1111,TE1112,TE1113,TE1114,TE1115,TE1116,TE1117,TE1118,TE1119,TE1120,TE1121,TE1122,TE1123,TE1124,TE1125,TE1126,TE1127,TE1128,TE1129,TE1130,TE1131,TE1132,TE1133,TE1134,TE1135,TE1136,TE1137,TE1138,TE1139,TE1140,TE1141,TE1142,TE1143,TE1144,TE1145,TE1146,TE1147,TE1148,TE1149,TE1150,TE1151,TE1152,TE1153,TE1154,TE1155,TE1156,TE1157,TE1158,TE1159,TE1160,TE1161,TE1162,TE1163,TE1164,TE1165,TE1166,TE1167,TE1168,TE1169,TE1170,TE1171,TE1172,TE1173,TE1174,TE1175,TE1176,TE1177,TE1178,TE1179,TE1180,TE1181,TE1182,TE1183,TE1184,TE1185,TE1186,TE1187,TE1188,TE1189,TE1190,TE1191,TE1192,TE1193,TE1194,TE1195,TE1196,TE1197,TE1198,TE1199,TE1200,TE1201,TE1202,TE1203,TE1204,TE1205,TE1206,TE1207,TE1208,TE1209,TE1210,TE1211,TE1212,TE1213,TE1214,TE1215,TE1216,TE1217,TE1218,TE1219,TE1220,TE1221,TE1222,TE1223,TE1224,TE1225,TE1226,TE1227,TE1228,TE1229,TE1230,TE1231,TE1232,TE1233,TE1234,TE1235,TE1236,TE1237,TE1238,TE1239,TE1240,TE1241,TE1242,TE1243,TE1244,TE1245,TE1246,TE1247,TE1248,TE1249,TE1250,TE1251,TE1252,TE1253,TE1254,TE1255,TE1256,TE1257,TE1258,TE1259,TE1260,TE1261,TE1262,TE1263,TE1264,TE1265,TE1266,TE1267,TE1268,TE1269,TE1270,TE1271,TE1272,TE1273,TE1274,TE1275,TE1276,TE1277,TE1278,TE1279,TE1280,TE1281,TE1282,TE1283,TE1284,TE1285,TE1286,TE1287,TE1288,TE1289,TE1290,TE1291,TE1292,TE1293,TE1294,TE1295,TE1296,TE1297,TE1298,TE1299,TE1300,TE1301,TE1302,TE1303,TE1304,TE1305,TE1306,TE1307,TE1308,TE1309,TE1310,TE1311,TE1312,TE1313,TE1314,TE1315,TE1316,TE1317,TE1318,TE1319,TE1320,TE1321,TE1322,TE1323,TE1324,TE1325,TE1326,TE1327,TE1328,TE1329,TE1330,TE1331,TE1332,TE1333,TE1334,TE1335,TE1336,TE1337,TE1338,TE1339,TE1340,TE1341,TE1342,TE1343,TE1344,TE1345,TE1346,TE1347,TE1348,TE1349,TE1350,TE1351,TE1352,TE1353,TE1354,TE1355,TE1356,TE1357,TE1358,TE1359,TE1360,TE1361,TE1362,TE1363,TE1364,TE1365,TE1366,TE1367,TE1368,TE1369,TE1370,TE1371,TE1372,TE1373,TE1374,TE1375,TE1376,TE1377,TE1378,TE1379,TE1380,TE1381,TE1382,TE1383,TE1384,TE1385,TE1386,TE1387,TE1388,TE1389,TE1390,TE1391,TE1392,TE1393,TE1394,TE1395,TE1396,TE1397,TE1398,TE1399,TE1400,TE1401,TE1402,TE1403,TE1404,TE1405,TE1406,TE1407,TE1408,TE1409,TE1410,TE1411,TE1412,TE1413,TE1414,TE1415,TE1416,TE1417,TE1418,TE1419,TE1420,TE1421,TE1422,TE1423,TE1424,TE1425,TE1426,TE1427,TE1428,TE1429,TE1430,TE1431,TE1432,TE1433,TE1434,TE1435,TE1436,TE1437,TE1438,TE1439,TE1440,TE1441,TE1442,TE1443,TE1444,TE1445,TE1446,TE1447,TE1448,TE1449,TE1450,TE1451,TE1452,TE1453,TE1454,TE1455,TE1456,TE1457,TE1458,TE1459,TE1460,TE1461,TE1462,TE1463,TE1464,TE1465,TE1466,TE1467,TE1468,TE1469,TE1470,TE1471,TE1472,TE1473,TE1474,TE1475,TE1476,TE1477,TE1478,TE1479,TE1480,TE1481,TE1482,TE1483,TE1484,TE1485,TE1486,TE1487,TE1488,TE1489,TE1490,TE1491,TE1492,TE1493,TE1494,TE1495,TE1496,TE1497,TE1498,TE1499,TE1500,TE1501,TE1502,TE1503,TE1504,TE1505,TE1506,TE1507,TE1508,TE1509,TE1510,TE1511,TE1512,TE1513,TE1514,TE1515,TE1516,TE1517,TE1518,TE1519,TE1520,TE1521,TE1522,TE1523,TE1524,TE1525,TE1526,TE1527,TE1528,TE1529,TE1530,TE1531,TE1532,TE1533,TE1534,TE1535,TE1536,TE1537,TE1538,TE1539,TE1540,TE1541,TE1542,TE1543,TE1544,TE1545,TE1546,TE1547,TE1548,TE1549,TE1550,TE1551,TE1552,TE1553,TE1554,TE1555,TE1556,TE1557,TE1558,TE1559,TE1560,TE1561,TE1562,TE1563,TE1564,TE1565,TE1566,TE1567,TE1568,TE1569,TE1570,TE1571,TE1572,TE1573,TE1574,TE1575,TE1576,TE1577,TE1578,TE1579,TE1580,TE1581,TE1582,TE1583,TE1584,TE1585,TE1586,TE1587,TE1588,TE1589,TE1590,TE1591,TE1592,TE1593,TE1594,TE1595,TE1596,TE1597,TE1598,TE1599,TE1600,TE1601,TE1602,TE1603,TE1604,TE1605,TE1606,TE1607,TE1608,TE1609,TE1610,TE1611,TE1612,TE1613,TE1614,TE1615,TE1616,TE1617,TE1618,TE1619,TE1620,TE1621,TE1622,TE1623,TE1624,TE1625,TE1626,TE1627,TE1628,TE1629,TE1630,TE1631,TE1632,TE1633,TE1634,TE1635,TE1636,TE1637,TE1638,TE1639,TE1640,TE1641,TE1642,TE1643,TE1644,TE1645,TE1646,TE1647,TE1648,TE1649,TE1650,TE1651,TE1652,TE1653,TE1654,TE1655,TE1656,TE1657,TE1658,TE1659,TE1660,TE1661,TE1662,TE1663,TE1664,TE1665,TE1666,TE1667,TE1668,TE1669,TE1670,TE1671,TE1672,TE1673,TE1674,TE1675,TE1676,TE1677,TE1678,TE1679,TE1680,TE1681,TE1682,TE1683,TE1684,TE1685,TE1686,TE1687,TE1688,TE1689,TE1690,TE1691,TE1692,TE1693,TE1694,TE1695,TE1696,TE1697,TE1698,TE1699,TE1700,TE1701,TE1702,TE1703,TE1704,TE1705,TE1706,TE1707,TE1708,TE1709,TE1710,TE1711,TE1712,TE1713,TE1714,TE1715,TE1716,TE1717,TE1718,TE1719,TE1720,TE1721,TE1722,TE1723,TE1724,TE1725,TE1726,TE1727,TE1728,TE1729,TE1730,TE1731,TE1732,TE1733,TE1734,TE1735,TE1736,TE1737,TE1738,TE1739,TE1740,TE1741,TE1742,TE1743,TE1744,TE1745,TE1746,TE1747,TE1748,TE1749,TE1750,TE1751,TE1752,TE1753,TE1754,TE1755,TE1756,TE1757,TE1758,TE1759,TE1760,TE1761,TE1762,TE1763,TE1764,TE1765,TE1766,TE1767,TE1768,TE1769,TE1770,TE1771,TE1772,TE1773,TE1774,TE1775,TE1776,TE1777,TE1778,TE1779,TE1780,TE1781,TE1782,TE1783,TE1784,TE1785,TE1786,TE1787,TE1788,TE1789,TE1790,TE1791,TE1792,TE1793,TE1794,TE1795,TE1796,TE1797,TE1798,TE1799,TE1800,TE1801,TE1802,TE1803,TE1804,TE1805,TE1806,TE1807,TE1808,TE1809,TE1810,TE1811,TE1812,TE1813,TE1814,TE1815,TE1816,TE1817,TE1818,TE1819,TE1820,TE1821,TE1822,TE1823,TE1824,TE1825,TE1826,TE1827,TE1828,TE1829,TE1830,TE1831,TE1832,TE1833,TE1834,TE1835,TE1836,TE1837,TE1838,TE1839,TE1840,TE1841,TE1842,TE1843,TE1844,TE1845,TE1846,TE1847,TE1848,TE1849,TE1850,TE1851,TE1852,TE1853,TE1854,TE1855,TE1856,TE1857,TE1858,TE1859,TE1860,TE1861,TE1862,TE1863,TE1864,TE1865,TE1866,TE1867,TE1868,TE1869,TE1870,TE1871,TE1872,TE1873,TE1874,TE1875,TE1876,TE1877,TE1878,TE1879,TE1880,TE1881,TE1882,TE1883,TE1884,TE1885,TE1886,TE1887,TE1888,TE1889,TE1890,TE1891,TE1892,TE1893,TE1894,TE1895,TE1896,TE1897,TE1898,TE1899,TE1900,TE1901,TE1902,TE1903,TE1904,TE1905,TE1906,TE1907,TE1908,TE1909,TE1910,TE1911,TE1912,TE1913,TE1914,TE1915,TE1916,TE1917,TE1918,TE1919,TE1920,TE1921,TE1922,TE1923,TE1924,TE1925,TE1926,TE1927,TE1928,TE1929,TE1930,TE1931,TE1932,TE1933,TE1934,TE1935,TE1936,TE1937,TE1938,TE1939,TE1940,TE1941,TE1942,TE1943,TE1944,TE1945,TE1946,TE1947,TE1948,TE1949,TE1950,TE1951,TE1952,TE1953,TE1954,TE1955,TE1956,TE1957,TE1958,TE1959,TE1960,TE1961,TE1962,TE1963,TE1964,TE1965,TE1966,TE1967,TE1968,TE1969,TE1970,TE1971,TE1972,TE1973,TE1974,TE1975,TE1976,TE1977,TE1978,TE1979,TE1980,TE1981,TE1982,TE1983,TE1984,TE1985,TE1986,TE1987,TE1988,TE1989,TE1990,TE1991,TE1992,TE1993,TE1994,TE1995,TE1996,TE1997,TE1998,TE1999,TE2000,TE2001,TE2002,TE2003,TE2004,TE2005,TE2006,TE2007,TE2008,TE2009,TE2010,TE2011,TE2012,TE2013,TE2014,TE2015,TE2016,TE2017,TE2018,TE2019,TE2020,TE2021,TE2022,TE2023,TE2024,TE2025,TE2026,TE2027,TE2028,TE2029,TE2030,TE2031,TE2032,TE2033,TE2034,TE2035,TE2036,TE2037,TE2038,TE2039,TE2040,TE2041,TE2042,TE2043,TE2044,TE2045,TE2046,TE2047,TE2048,TE2049,TE2050,TE2051,TE2052,TE2053,TE2054,TE2055,TE2056,TE2057,TE2058,TE2059,TE2060,TE2061,TE2062,TE2063,TE2064,TE2065,TE2066,TE2067,TE2068,TE2069,TE2070,TE2071,TE2072,TE2073,TE2074,TE2075,TE2076,TE2077,TE207
```


C <DCL10>DBM16F.FOR.1 Mon 18-Nov-88 6:38PM

C DBM16F: REAL TIME SYSTEM LOOP FUNCTION LIST.
C
CALL MWPL(BMPL)
CALL MPIFF(TSRPA,TOTPA,ANLZA)
CALL MPIFF(RSTPA,RSTPA,SYNZA)
CALL MPIFF(TSRPA,TOTPA,ANLZA)
CALL MPIFF(RSTPA,RSTPA,SYNZA)
CALL MWPL(BMPL)

~L

C <DCL10>DBM16F.FOR.1 Mon 18-Nov-88 6:38PM

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

C DEFINE UPPER LEVEL SYSTEM FUNCTION LISTS:

C DBM16F, BMPL, BMPLT, BMPLT, BMPLT, BMPLT

C DBM16F, BMPL, BMPLT, BMPLT, BMPLT, BMPLT

C DBM16F: REAL TIME SYSTEM STARTUP FUNCTION LIST.

C

CALL MWPL(BMPLT)

CALL MPIFF(TSRPA,TOTPA,ANLZA)

CALL MPIFF(RSTPA,RSTPA,SYNZA)

CALL MPIFF(TSRPA,TOTPA,ANLZA)

CALL MPIFF(RSTPA,RSTPA,SYNZA)

CALL MWPL(BMPLT)

CALL MWPL(BMPLT)

~L

C <DCA16>RBM16F.FOR.1 Mon 18-Nov-80 6:38PM

C BMWFT: FILE-TD-FILE SYSTEM FUNCTION LIST.

C
CALL MWFL(BMWFT)
CALL MWFT(PCSR,AP)
CALL ADINT
CALL RMINT
CALL MPIFF(TSRFA,TBTF,ANLZA)
CALL MPIFF(RBTF,RSNFA,SYNZA)
CALL MPIFF(TSRFB,TBTF,ANLZH)
CALL MPIFF(RBTF,RSNFB,SYNZB)
CALL MPAT(PCSM,AP)
CALL TMINT
CALL DAINT
CALL MWFL(BWFFT)

-L

C <DCA16>RBM16F.FOR.1 Mon 18-Nov-80 6:38PM

C BMWST: REAL TIME SYSTEM RESTANT FUNCTION LIST.

C
CALL MWFL(BMWST)
CALL MPIS(RUN,1)
CALL RMIM(RUN,ME,M,BMMLP)
CALL MWFL(BWST)

-L

С <UCJ16>M167.YO.1.1 MON 16-MAY-06 13:41P R HPD

```

C ***** TIRING SYSTEM FUNCTION LIST.
C ***** NOTE: THIS FUNCTION LIST INCLUDES ONLY THE
C ***** SIGNAL PROCESSING PORTIONS OF THE SYSTEM.
C ***** IT DOES NOT REPRESENT A FUNCTIONING SPEECH CODING
C ***** SYSTEM. THE FOLLOWING ROUTINES ARE NOT INCLUDED:
C ***** ADINT, TINT, MINT, DINT.
C *****
C ***** CALL MPFL(DUNITM)
C ***** CALL MPFL(AUNITI)
C ***** CALL MPFL(SANZA)
C ***** CALL MPFL(SUNITM)

```

7

C <DCAL6>RRN10F.FUR.1 MON 10-NOV-68 6:38PM Page 9

```

C *****
C DEFINE LOWER LEVEL FUNCTION LISTS:
C  AMLZ1, AMLZ2, SYNZA, SYNZR
C
C  AMLZA: ANALYSIS FROM TSRA -> TSMKA)
C      PRCTCT FROM TSMKW -> TBTB. (PROPAR & PROGRES)
C
C      CALL MPRPL(AMLZA)
C
C  C SET G3 FOR SCOPE SYNC AT START OF TRANSMITTER
C      CALL MPGSC(IG3,ISRT)
C
C  C MONITOR PEAK INPUT LEVEL
C      CALL ADPEAK(TSRA)
C
C  C TRANSMITTER FRAME HISTORY BUFFER UPDATES...
C      CALL THIST(TSP1,TSP2,TE1,TE11,TRU,TRHW)
C      CALL MPRBF(214)
C
C  C FREQUENCY BASIS....
C      CALL DPRT(TSP2,TALPH,TSRA,TPHST,TUNES)
C      CALL MPRBF(215)
C
C  C LONG WITH TSMW....
C      CALL MPWT(PRCSE,AP)
C      CALL MPIST(TSRA,0)
C
C  C PITCH FILTER ANALYSIS....
C      CALL SSUM(TDCM,TSP,TFSZ1)
C      CALL MPRBF(217)
C      CALL WHUL(TDSP,1,TSP,TDCM,THANG,0)
C      CALL MPRBF(218)
C      CALL PFTLR(TSP,1,TWSP2,TCUST,TXSP)
C      CALL MPRBF(219)
C      CALL MPRBF(220)
C      CALL WMSM(TPSPM,TXSP,TXSP1)
C      CALL MPRBF(221)
C      CALL VCLR(TPSP1)
C      CALL MPRBF(222)
C      CALL PFLR(TRP,1,TPSP,TCUST,TRP)
C      CALL MPRBF(223)
C      CALL MPRBF(224)
C      CALL PITCH(TPPP,TMH,TRP,TIM,T14,TC2P)
C      CALL MPRBF(225)
C      CALL MEWC(TC,TINC,TRPP,TKTHH,TRP,TWORK)
C      CALL MPRBF(226)
C      -CALL STAB(TCM,TC2P,TIC,TC)
C      CALL MPRBF(227)
C
C  C INVERSE PITCH FILTER...

```



```

C <UCAL6>BUN10F.FUN.1 Mon 18-Nov-88 6:38PM Page 11
C <UCAL6>BUN10F.FUN.1 Mon 18-Nov-88 6:38PM Page 11:1

C CALL VML(TWK1,1,TX10,0,TWAM1,0)
C CALL MPBF(229)
C CALL DCONZ(TNS,0,TWK1,TWK1)
C CALL MPBF(230)
C CALL WPC(TSP,TNS,TLM)
C CALL MPBF(231)
C CALL MLQ(TK,TXNR,TM,TNSP,TIK)
C CALL MPBF(232)

C INVERSE SPECTRAL FILTER...
C CALL VADN(TAM,TEN)
C CALL MPBF(233)
C CALL DCONZ(TZ,0,TZ1,TAMR)
C CALL MPBF(234)

C BMM PROTECTION ON PREVIOUS FRAME PARAMETER DATA
C (HIDDEN UNDER DCONZ FUNCTION)...
C CALL PROP(A)

C NOISE SHAPING FILTER CALCULATION...
C CALL VML(TAM,1,TAM,0,TZAC,0)
C CALL MPBF(235)

C GAIN FACTOR CALCULATION...
C CALL GAIN(TCFAC,TIC,TIDC,TZ2,TZ3,T3R)
C CALL MPBF(236)

C APC RESIDUAL CALCULATION ("TUB" IS EQUIVALENT TO START OF "TSHK")...
C CALL APC(TUB,TM,TSP,TZ1,TZ2,TZ3,T3R,T3I)
C CALL MPBF(238)

C ERROR PROTECTION ON PREVIOUS FRAME RESIDUAL DATA
C (HIDDEN UNDER APC FUNCTION)...
C CALL PHRES(A)
C CALL MPBF(239)

C TRANSMITTED DATA COLLECTION...
C CALL GTH(TSHK,TIC,TIDC,TM,TIC,TIK)
C CALL MPBF(240)

C CALL MPBF(AMZB)

C
~L

C <UCAL6>BUN10F.FUN.1 Mon 18-Nov-88 6:38PM Page 11:1
C <UCAL6>BUN10F.FUN.1 Mon 18-Nov-88 6:38PM Page 11

C SYNZA: SYNTHESIS FROM RSRA -> KSKA)
C DECODE/COMPAR FROM RTU -> KSHB.
C CALL MPBF(SYNZA)

C CALL MPSC(103,ICLR)

C RECEIVER DATA DISTRIBUTION...
C CALL DEAL(RSRA,RCH,ROCH,RNH,RCH,RKH)
C CALL MPBF(241)

C RECEIVER FRAME HISTORY BUFFER UPDATE...
C CALL VMOV(RRH,RRH)
C CALL MPBF(243)

C RESIDUAL SCALING ("RMA" IS EQUIVALENT TO START OF "RSRA")...
C CALL SCLRES(RMH,RCH,ROCH,RUHA)
C CALL MPBF(244)

C SPECTRAL SYNTHESIS...
C CALL VLSY(RMH,RNH,RKH,RKH)
C CALL MPBF(246)

C DECODING OF PREVIOUS FRAME RECEIVER RESIDUAL DATA &
C ERROR CORRECTION ON PREVIOUS FRAME RECEIVER PARAMETER DATA
C (PARTIALLY HIDDEN UNDER VLSY FUNCTION)...
C CALL DECOD
C CALL CORPAR(H)
C CALL MPBF(PDCSH,AP)
C CALL MPBF(ROTFB,0)

C PITCH SYNTHESIS...
C CALL PITSY(RMH,RNH,RKH,RCH)
C CALL MPBF(247)

C DEEMPHASIS ("RRHA" IS EQUIVALENT TO "RSKA")...
C CALL DFL22(RRHA,RCH,ROCH,RNH,RKH,RKH)
C CALL MPBF(248)

C KSKA IS FULL...
C CALL MPBF(PDCSH,AP)
C CALL MPBF(RSNVA,1)

C CALL MPBF(SYNZA)

~L

```

C
RETURN
END

C SYMB: SYNTHESIS FROM RSNB -> RSNKB
C DECOM/COMPAR FROM DATA -> RSRA.
C
C CALL MPBFL(SYNZB)
C
C CALL MPSC(IG3,ICLN)
C RECEIVER DATA DISTRIBUTION...
C CALL DEAL(RSNB,RCH,RDCH,RNH,RCH,RKH)
C CALL MPBFL(242)
C RECEIVER FRAME MISTERY BUFFER UPDATE...
C CALL VMDW(RNH,RNH)
C CALL MPBFL(243)
C RESIDUAL SCALING ('RNB' IS EQUIVALENT TO START OF 'RSRB')...
C CALL SCAL(RNH,RCH,RDCH,RNH)
C CALL MPBFL(245)
C SPECTRAL SYNTHESIS...
C CALL VTSY(RNH,RNH,RKH,RNH)
C CALL MPBFL(246)
C DECODING OF PREVIOUS FRAME RECEIVER RESIDUAL DATA &
C ERROR CORRECTION ON PREVIOUS FRAME RECEIVER PARAMETER DATA
C (PARTIALLY MISTERY UNDER VTSY FUNCTION)...
C CALL DECUA
C CALL COMP(1A)
C CALL MPWT(PRC5B,AP)
C CALL MPST(RNCPA,B)
C PITCH SYNTHESIS...
C CALL PITST(RNH,RNH,RNH,RNH)
C CALL MPBFL(247)
C RESYNTHESIS ('RNB' IS EQUIVALENT TO 'RSNB')...
C CALL OFL22(RNDB,RNCPB,RNH,RNH)
C CALL MPBFL(249)
C RSNB IS FULL...
C CALL MPWT(PRC5B,AP)
C CALL MPST(RNCPB,1)
C CALL MPBFL(SYNZB)

-L

```

C  <UCLAL>B8MINH.FUR.1  Non 10-Nov-88  5:40PM
C
C
C
C  INTEGER FUNCTION ADINT(I)
C
C  FCB # 124
C  ADINT=INTPM(124,I)
C  RETURN
C  END
C
C
C
C  INTEGER FUNCTION ADPEAK(U)
C
C  USAGE: IER = ADPEAK(U)
C
C  INTEGER U,FCBCH
C  ADPEAK = FCBCH(102,0,0,U,0,0,0,0,0,10)
C  RETURN
C  END
C
C
C
C  INTEGER FUNCTION APC(V,A,U,V,W,R,S,T)
C
C  USAGE: IER = APC(V,A,U,V,W,R,S,T)
C
C  NOTE: V,U,V,W,R,S,T ARE BUFFERS; A IS SCALAR
C  FUNCTION REQUIRES SPECIAL BINDING
C
C  INTEGER V,A,U,V,W,R,S,T,FCBCH
C  APC = FCBCH(-199,V,A,U,V,W,R,S,T,14)
C  RETURN
C  END
C
C
C
C  INTEGER FUNCTION CORPAR(IAB)
C
C  IAB=-2 -> CORRECT MATA INTO MSRA
C  IAB=0 -> CORRECT MATA INTO MSRH
C  FCB #123
C
C  COMPAR=INTPM(123,IAB)
C  RETURN
C  END
C
C
C
C  INTEGER FUNCTION DADINT(I)

```

```
C <DCALIB>>BUNION.FOR-1 Mon 18-Nov-80 01:49PM Page 1
C (*****DCALIB>BUNION.FUN-1, 18-NOV-80 10:40:30, Ed: KFIELD
C *****)
C
C THIS FILE CONTAINS MAP-300 SHAPII MOST SUPPORT FUNCTIONS
C USED BY THE NEW DCAL6 SYSTEM.
C
C FUNCTIONS ARE ORDERED ALPHABETICALLY.
C
C
C
C
C
```



```

C
C
C
      INTEGER FUNCTION MPGSC(GFLAG,SETCLR)
C
C     SUPPORT MODULE FOR MPGSC -- GFLAG SET/CLEAR
C
C     CALLING SEQUENCE:
C     IER = MPGSC(GFLAG,SETCLR)
C     VALUES:
C     GFLAG = G-FLAG NUMBER (0,1,2,3)
C     SETCLR = 0 TO CLEAR IT, 1 TO SET IT
C
C     INTEGER GFLAG,SETCLR
C     INTEGER FCBGC,FCBSZ,MUS,FCB,RHUMP,HRI,FLID
C     COMMON/MP22Z/FCBGC(11),FCBSZ(11,7),MUS,FCB(6),MPDCB(4),HRI,LEVEI
C
C     MPGSC = 0
C
C     DO 10 I=1,11
C       FCBGC(I) = 0
C   10
C     FCB B106
C
C     FCBGC(2) = 106
C
C     IF ((SETCLR .LT. 0) .OR. (SETCLR .GT. 1)) MPGSC = -2
C     IF ((GFLAG .LT. 0) .OR. (GFLAG .GT. 3)) MPGSC = -1
C
C     IF (MPGSC .NE. 0) CALL MPHERR(MPGSC)
C     IF (MPGSC .NE. 0) RETURN
C
C     FCBGC(5) = GFLAG
C     FCBGC(7) = SETCLR
C
C     MPGSC = RHUMP(FCBGC(1),FCBSZ(1,5))
C
C     RETURN
C     END
C
C
C
      INTEGER FUNCTION MPFF(ISA,ISB,PLID)
C
C     SUPPORT MODULE FOR THE INTEGER "IF A.NE.B .AND. B.EQ.Q"
C     CONDITIONAL FCN.
C
C     CALLING SEQUENCE:
C     IER = MPFF(ISA,ISB,PLID)
C     VALUES:
C     ISA = INTEGER SCALAR ID (0 .LE. 15A .LE. 127)

```

C <UCALB>BUN16H.FOR.1

Mon 18-Nov-88 6:48PM

Page 2:7

C <UCALB>BUN16H.FOR.1

Mon 18-Nov-88 6:48PM

```

C
C      INTEGER V,A,U,B,C,D,PCBCH
C      IOUT=V
C      PITCH=PCBCH(134,V,A,B,C,D,PCBCH,C,IOUT,B,14)
C      RETURN
C      END
C
C      INTEGER FUNCTION PITCH(V,A,U,V)
C
C      MOST SUPPORT FOR PITCH
C      USAGE: IERR=PITCH(V,A,U,V)
C
C      INTEGER V,A,U,V,PCBCH
C      PITCH=PCBCH(203,V,A,B,C,D,V,B,B,B,9)
C      RETURN
C      END
C
C      INTEGER FUNCTION PROPAB(IAB)
C
C      IAB--2 => PROTECT (PARAMETERS) TSKA INTO TOTA
C      IAB--0 => PROTECT (PARAMETERS) TSKB INTO TOTA
C      PCB #122
C
C      PROPAB=INTFN(122,IAB)
C      RETURN
C      END
C
C      INTEGER FUNCTION PROMES(IAM)
C
C      IAB--2 => PROTECT (RESIDUAL) TSKA INTO TOTA
C      IAB--0 => PROTECT (RESIDUAL) TSKB INTO TOTA
C      PCB #121
C
C      PROMES=INTFN(121,IAB)
C      RETURN
C      END
C
C      INTEGER FUNCTION RMINT(I)
C
C      PCB # 126
C      RMINT=INTFN(126,I)
C      RETURN
C      END
C
C      INTEGER FUNCTION SCLRES(V,A,U,V)
C
C      MOST SUPPORT FOR SCLRES
C      USAGE: IERR=SCLRES(V,A,U,V)
C
C      INTEGER V,A,U,V,PCBCH
C      SCLRES=PCBCH(202,V,A,U,B,B,B,B,B,9)
C      RETURN
C      END
C
C      INTEGER FUNCTION STAB(V,A,U,V)
C
C      MOST SUPPORT FOR STAB
C      USAGE: IERR=STAB(V,A,U,V)
C
C      INTEGER V,A,U,V,PCBCH
C      STAB=PCBCH(150,V,A,U,B,B,B,B,B,9)
C      RETURN
C      END
C
C      INTEGER FUNCTION THIST(V,U,V,M,R,S)
C
C      USAGE: IER = THIST(V,U,V,M,R,S)
C
C      NOTE: ALL AMGS ARE BUFFERS: SPECIAL HANDLING IS REQ'D.
C
C      INTEGER V,U,V,M,R,S,PCBCH
C      THIST = PCBCH(-241,V,U,V,M,R,S,B,B,3)
C      RETURN
C      END
C
C      INTEGER FUNCTION TINT(I)
C
C      PCB # 125
C      TINT=INTFN(125,I)
C      RETURN
C      END
C
C      INTEGER FUNCTION RETOA(V,B)

```

```

C <DEAL>SUBROUTINE.FOR-1 MON 18-MAY-80 6:48PM
C MUST SUPPORT FOR VETUA
C USAGE: IERR=VETUA(V,B)
C
C INTEGER V,B,PCBCH
C
C VETUA=PCBCH(13,V,B,B,B,B,B,B,11)
C RETURN
C
C
C
C INTEGER FUNCTION VETSV(V,B,V,B)
C
C USAGE IER = VETSV(V,B,V,B)
C
C NOTE: USE CALLING SEQUENCE 017 (V,B,V,B), WITH A=0
C
C INTEGER PCBCH,V,B,V,B
C VETSV = PCBCH(132,V,B,B,B,B,B,B,17)
C RETURN
C

```

[illegible]


```

C <MAP>HIST16.FOR.7   Tue 30-Dec-80 9:27AM           Page 1

C (ENDMD)MAP>HIST16.FOR.7, 30-Dec-80 09:27:12, 843 MILE
C HIST16 --> PRINT OUT HISTOGRAM DATA FROM DCA16 SPEECH CODEX

      INTEGER HISTO,FID,CHTC,LNG,BUS1
      INTEGER M(260),G(64),K(64),C1(2),C2(16),C3(8)
      INTEGER K1(64),K2(32),K3(16),K4(16),K5(16),K6(16)
      EQUIVALENCE (M(1),G(1)),(M(65),G(1)),(M(89),C1(1)),
1 (M(77),C2(1)),(M(93),C3(1)),(M(101),K1(1)),(M(165),K2(1)),
2 (M(197),K3(1)),(M(213),K4(1)),(M(229),K5(1)),(M(245),K6(1))
      DATA HISTO,BUS1,FID,CHTC,LNG/1,64,2,1,8/
      DATA HISTO,SHISTO/39944.,260./

      CALL WPUH(3)
      TYPE 1
      FORMAT(' FILE (OR TT) FOR HISTOGRAM: 'S)
      CALL ASSIGN(2,M(1),-1)
      CALL WPLD(BUS1+HISTO,SHISTO,SHISTO,FID,CHTC,LNG)
      CALL WPMU(HISTO,M(1),2,0,N(260))
      NPN=0
      DO 5 I=1,4
10      WPM=HFM+NG(I)
      WRITE(2,10) WPM
      FORMAT(' TOTAL FRAMES = ',16//,' M 6 DG C1 C2',
1 - C3 K1 K2 K3 K4 K5 K6//)
      DO 15 I=1,4
15      J=1-1
      WRITE(2,16) J,G(1),G(1),C1(1),C2(1),C3(1),K1(1),K2(1),
1 K3(1),K4(1),K5(1),K6(1)
      FORMAT(13,1116)
      DO 20 I=5,8
20      J=1-1
      WRITE(2,21) J,G(1),C1(1),C2(1),C3(1),K1(1),K2(1),
1 K3(1),K4(1),K5(1),K6(1)
      FORMAT(13,16,61,916)
      DO 25 I=9,16
25      J=1-1
      WRITE(2,26) J,G(1),C2(1),K1(1),K2(1),
1 K3(1),K4(1),K5(1),K6(1)
      FORMAT(13,16,61,16,61,716)
      DO 30 I=17,32
30      J=1-1
      WRITE(2,31) J,G(1),K1(1),K2(1)
31      FORMAT(13,16,4(61),216)
      DO 35 I=33,64
35      J=1-1
      WRITE(2,36) J,G(1),K1(1)
36      FORMAT(13,16,4(61),16)
      WRITE(2,40)
      FORMAT(//)
      CALL CLOSE(2)
      STOP
      END

```

DATE
FILMED
-8