A REAL TIME IONOSPHERIC DRIFT SYSTEM

Description and operating
manual
(4 antennas)

by

C.E. MEEK
and
J. SOIFERMAN

Institute of Space and Atmospheric Studies
Physics Department
University of Saskatchewan
Saskatoon, Canada

-- March 1979 --

ERRATA (JUNE 1979)

During the second printing of this report, the opportunity has been taken to correct some minor errors in the text. Additional modifications which could not be easily implemented are listed below.

CH.2(Pg.5)

Reference should have been made to G.J.Fraser (PhD thesis, 1965, Univ. of Canterbury, N.Z.) who appears to have been the first to apply the binary correlation method to drift analysis.

CH.3(Pg.8)

The gate separation of 20µsec. corresponds to a free space sampling separation of 3Km.

(Pg.11,last paragraph)

P.I.A. is later used as an abbreviation for peripheral interface adaptor.

APP.A(Pg.40)

The first statement in interrupt should have been LDA $45 since there is some internal processing which stores the accumulator in $45 before the program jumps to the interrupt address. In the present program this omission is unimportant.

APP.B(Pg.62,line 4)

Insert "when they are not" after "(i.e. equal in)"

APP.D(Pg.82)

The delay parameter (CLKIN) given in the program actually covers the clocking in of 65 height gates.

Acknowledgements

The authors would like to thank Dr. D.G. Stephenson for his helpful comments on Chapter 3, and Mrs. Alice Brown for some of the typing and artwork.

Table of contents

Table of contents - continued

I. Introduction

The measurement of winds in the mesosphere by partially reflected radio waves consists in estimating the velocity of the ground amplitude pattern, which itself may be varying internally in time and have 'non-uniform' structure. Although there has been some argument in the past over the relation of this pattern velocity to the actual neutral drift, it is usually taken to be twice the drift velocity. The pattern is sample at at least three non-colinear ground locations (antennas) and at heights (given by the delay from the transmitted pulse) in the range ∼60-120 Km. A simple estimate of velocity, called apparent velocity, can be made by examining fading sequences at different antennas for similar features, and measuring their delay between antennas; or by calculating the cross correlations between fading sequences and looking for the lag at which there is maximum correlation (Fig. 1). This is a good estimate of the pattern velocity if the pattern, considered on the average over the record length, is statistically the same in all directions, and does not vary internally as it moves.

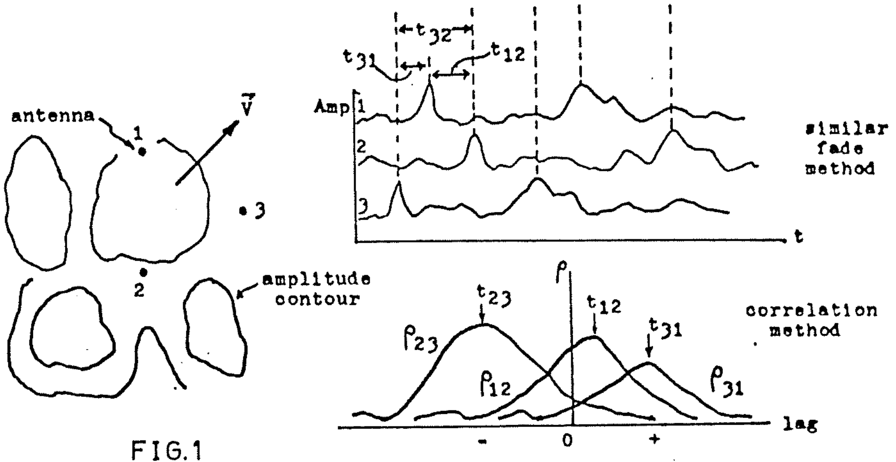


FIG.1

There is no way of testing for these conditions with just the time delays between antennas found above. It can be shown (Phillips and Spencer,1955) that if the pattern has a preferred direction of elongation not $\perp$ to the drift velocity, the measured apparent velocity has a tendency to be perpendicular to the direction of elongation and have a magnitude less than the actual drift (as shown in Fig. 2).

An extreme example of this would be a plane wave pattern.

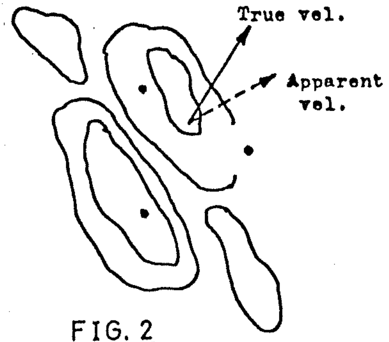


FIG. 2

If the pattern changes internally with time, an additional decrease in correlation with time delay (lag) between antennas, not related to the average pattern structure or drift velocity, shifts the positoon of maximum correlation towards zero lag - thus reducing the measured delays and increasing the apparent velocity (Fig. 3). This effect usually predominates, so that the apparent velocity is usually greater than the 'true' velocity -which has taken account of these possibilities.

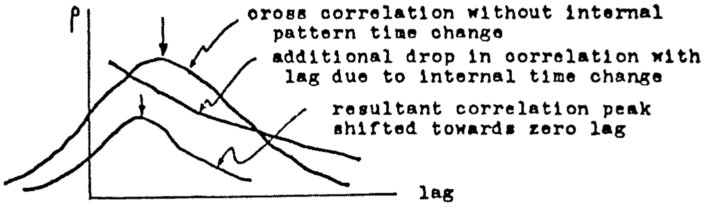


FIG.3

The rest of this discussion will be restricted to a three antenna (usually equilateral triangle) receiving array, or a system which simulates such an array (Fig. 4). It is fairly obvious that for a single fading feature, defined by the time in each of the three fading sequences, the sum of the delays (i.e. time differences) between antennas must be zero. This is also true when correlations are used to find the delays, assuming that the peaks selected are due to a single pattern motion, and can be shown to hold in the more general case of preferred directions of elongation and internal pattern time change (Meek,1978). This provides a valuable criterion for deciding whether peaks chosen from the cross correlations are the result of a pattern motion to which the theory applies. In general, if t_{ij} represents the

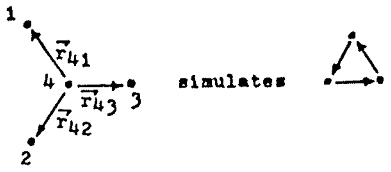


FIG.4

delay between antennas i and j (+ve if the pattern moves from i to j) with separation vector $\vec{r}_{ij}$; then if $\sum \vec{r}_{ij} = 0$, $\sum t_{ij}=0$. (The latter is not restricted by the shape of the array or the number of antennas). The normalized time discrepancy, NTD, will be defined by:

$$\mathrm{NTD} = \frac{\left|\sum t_{ij}\right|}{\sum \left|t_{ij}\right|}$$

and has a range 0-1.

Distributions of this parameter can be used to compare the data quality at different locations - i.e. how often are 'spurious' peaks the largest in the cross correlations. Since spurious peaks are quite common at our location, careful examination of the cross correlations is required to maximize data output while rejecting most of the spurious values. In the present analysis selection of peaks is based wholly on the NTD, although the widths of the peaks might also be useful (Meek,1978, Ch. 3).

There are many reasons for having real time analysis of data. Foremost is cost. For example, on the present university computer (IBM370), it takes ~1 HR to analyse a day's data (records every 5 min). A high record rate is desirable in order that tidal components, gravity waves, etc. may be examined. Although a more suitable computer would probably be cheaper in terms of execution time, the expense is still prohibitive. Second is storage. At the above rate, a standard 2000' tape (800BPI, recording the amplitude sequences) lasts about $1\frac{1}{2}$ days. Full (or partial) analysis of the data to be stored reduces this drastically - a factor of ~ 50 with the present system; i.e. a 2000' tape should last about 2 months. Third, since the data is being analysed in situ, the results may be fed back into the system control to change operating parameters, specifically receiver gain, so that data are collected under optimum conditions.

II. Considerations for a micro-computer system

There are three aspects to be considered; firstly the control of data gathering, secondly data correlations, and thirdly the calculation of drift velocities.

a) Control

In order that fading sequences may be correlated (without interpolation), pulses must be evenly spaced for each receiving antenna. A sample spacing of ∽0.5 sec at each antenna is a recommended maximum (∽150m antenna spacing) to insure that at least one cross correlation peak be far enough away from zero lag for accurate determination of, for example, NTD, given normally occuring wind speeds. Since timing is important here, this process should be under interrupt control by an external synch pulse which also controls the transmitter.

In addition to pulse timing, the signal must be sampled at an accurate and constant set of time delays. These "height gates" will be required to be at least as closely spaced as the transmitter pulse width (e.g. 20μsec in our case) to get the maximum height resolution (any smaller spacing means that adjacent height samples cannot be considered independent). Since micro-processor instructions take from ∽2-7μsec, it is rather difficult to put these gates under direct software control (although the delay to the first height gate, ∽50 Km, may be).

The standard system, in which raw fading sequences are recorded on magnetic tape and processed in a large computer, has the whole record (e.g. 300-400 amplitudes per height per antenna) available for analysis. This is impossible in a micro-computer due to storage limitations, and data must be dealt with in smaller portions, called blocks here. A block consists of (n amps.)x(m ants.)x(p hts.) amplitudes. It is easiest to use one byte (0-255) per amplitude.

b) Correlations

These must be done on a 'partial' basis,i.e. only one or two blocks available at a time. It is possible to do standard correlations by accumulating sums, squares, and cross products of amplitudes from two sequential small blocks whose minimum length is the maximum lag required; however, multiplication is very time consuming. For example, a test of this process on a mini-computer (BASIC) indicated that processing time (for 24 heights) was a factor of ∽ 36 greater than data collection time.

One solution, and that which is used presently, is to convert each block to binary with respect to the mean of the particular sequences. Two blocks of binary (which occupy $\frac{1}{4}$ the storage of an original block) are saved so that 'partial' correlations can be accumulated. This is very fast, since a single 'AND' instruction can be used to correlate 8 points. Another advantage of this process is that short-lived jumps in amplitude (Fig. 1) whose effects are often seen in the range ∽ 85-100 Km, and are probably due to meteor trails, only affect a small part of the whole record. A disadvantage is that if there are only a few scatterers distributed throughout the record (Fig. 2), the blocks recorded in between, essentially noise, will have equal effect on the correlations as the blocks with signal.

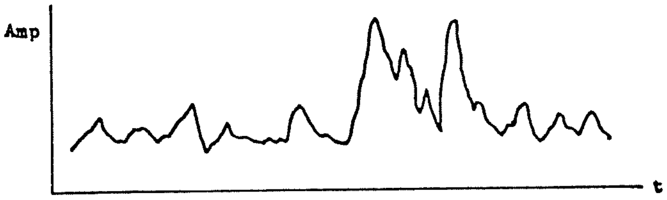


FIG. 1

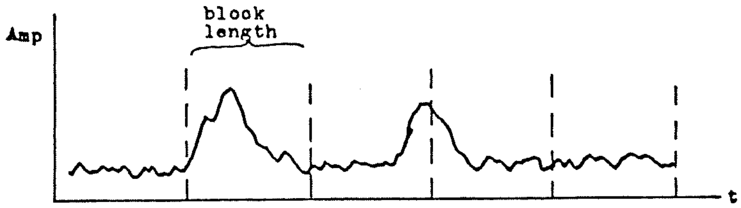


FIG. 2

Another solution (' average cross correlation') tested was to correlate each block separately and average (sum) the results. The test gave approximately the same number of wind values as the method above. Conceptually it seems that this would give more accurate values if the data were not stationary between blocks as shown in Fig. 3 (i.e. the mean block amplitude varies significantly). It would also be possible to reject blocks in which there was insufficient signal (i.e. noise). However, this would be no help at night when the signal to noise level is low - also variable rejection criteria would probably be required in the subsequent analysis depending on the number of blocks finally used at a given height.

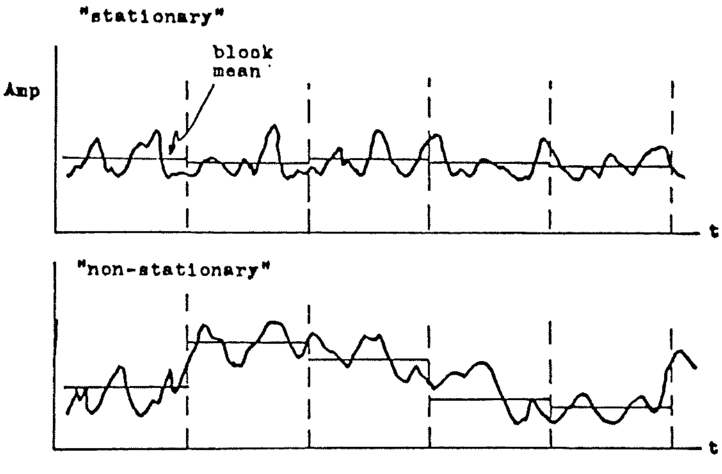


FIG.3

The latter comment raises an interesting point- viz. do the magnitudes of the correlation peaks depend on the length of the record. If the peaks are due to a constant pattern motion, then it appears that they should not be affected. However, spurious high correlation values are more likely, for statistical reasons, as the number of points correlated is reduced. Another interesting question involves the effect of block length on the results. It appears that the block method used constitutes a form of high pass filter on the raw amplitudes. Further investigation of these points would be worthwhile.

Because storage is limited, blocks must be converted to binary as soon as they are taken. In order that there not be a gap in the time sequences (although this is not **important** for the average cross correlation method), the conversion must be done between the handling of pulses. The length of time required depends on the block size. The available time depends on the pulse rate. A large block size is important for 'stationarity' reasons. As mentioned previously, choice of pulse rate is restricted by the required sample spacing.

Apart from binary conversion, there is a lot of free time left during a record which can be used for correlations and other tasks.

c) Calculation of wind

The determination of drift velocity is a 'numerical' rather than 'logical' process, and so is easiest to do in one of the higher computer languages.

This could be done in the same micro-computer, but would require a lot of time at the end of a record - and a good knowledge of the particular compiler or interpreter (e.g. BASIC) so that memory locations do not conflict. The easiest way is to use a second micro-computer to do the numerical computations; this also will allow a faster record rate, since the first computer will not have to pause for numerical calculations. The method of analysis must be fast, while still allowing 'quality' testing, and rejection of poor data. The latter task essentially consists in picking out the 'proper' peaks in the cross correlations. After the peaks have been found, there are several methods of analysis available. One is graphical (Briggs,1968), requiring the determination of crossing points of correlation curves, Another is a least squares fit to a Gaussian correlation function (Fedor,1967), which requires the inversion of a 6x6 matrix plus several other matrix operations. The method finally chosen was derived by Meek(1978), and is called the Poor man's full correlation analysis; poor not in the sense of giving poor results, but in that it requires the minimum amount of input data to produce a wind value. The graphical analysis is not suitable for our location because of our often irregularly shaped correlation peaks, and the least squares fit seems to require too much storage and execution time.

III. Brief description of present system and operation

a) Present system

Table 1 lists the present operating parameters. Fig. 1 is a plan of the transmitting and receiving arrays. Parallel dipoles in the outer squares are connected in parallel. The linear N-S mode is used for transmission and reception.

Table 1.

Transmitter:	power- 40 Kw
	pulse width- 20 μsec (variable)
	pulse rate- 7.5 Hz
	frequency- 2.22 MHz
	antenna ½ power beamwidth- ~22°
Receiver:	bandwidth- ~75 KHz
	Gain- voltage controlled, range ~60 dB
System:	Rx gain control- any of 8 manually set voltages
	# blocks/record- 8
	block length- 34 sec
	#height gates- 64 (32 used)
	gate width- 20 m
	gate separation- 20 μsec (programmable)
	record rate- every 5 min
	sampled height range- ~49-142 Km

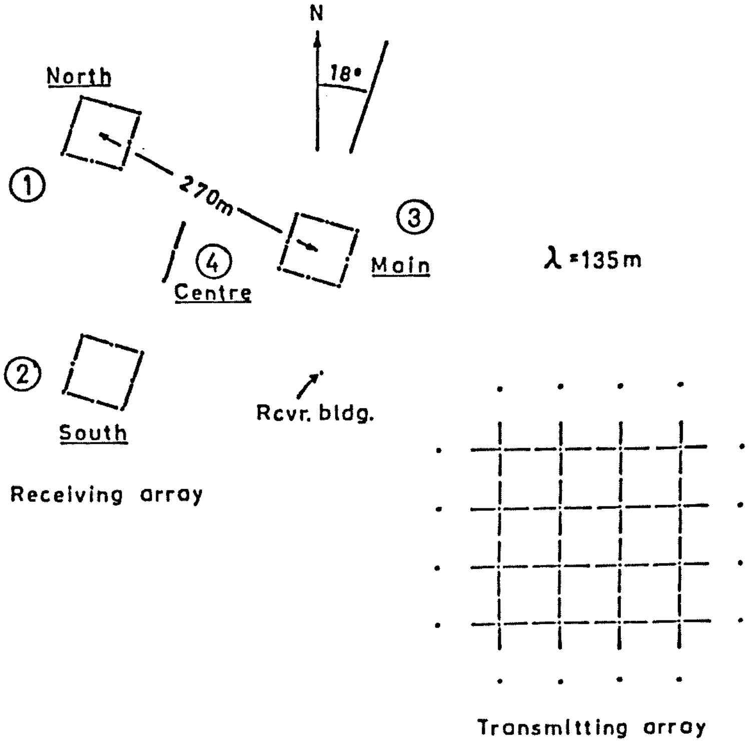


Fig. 1

Fig. 2 shows a block diagram of the receiving/analysing/recording system.

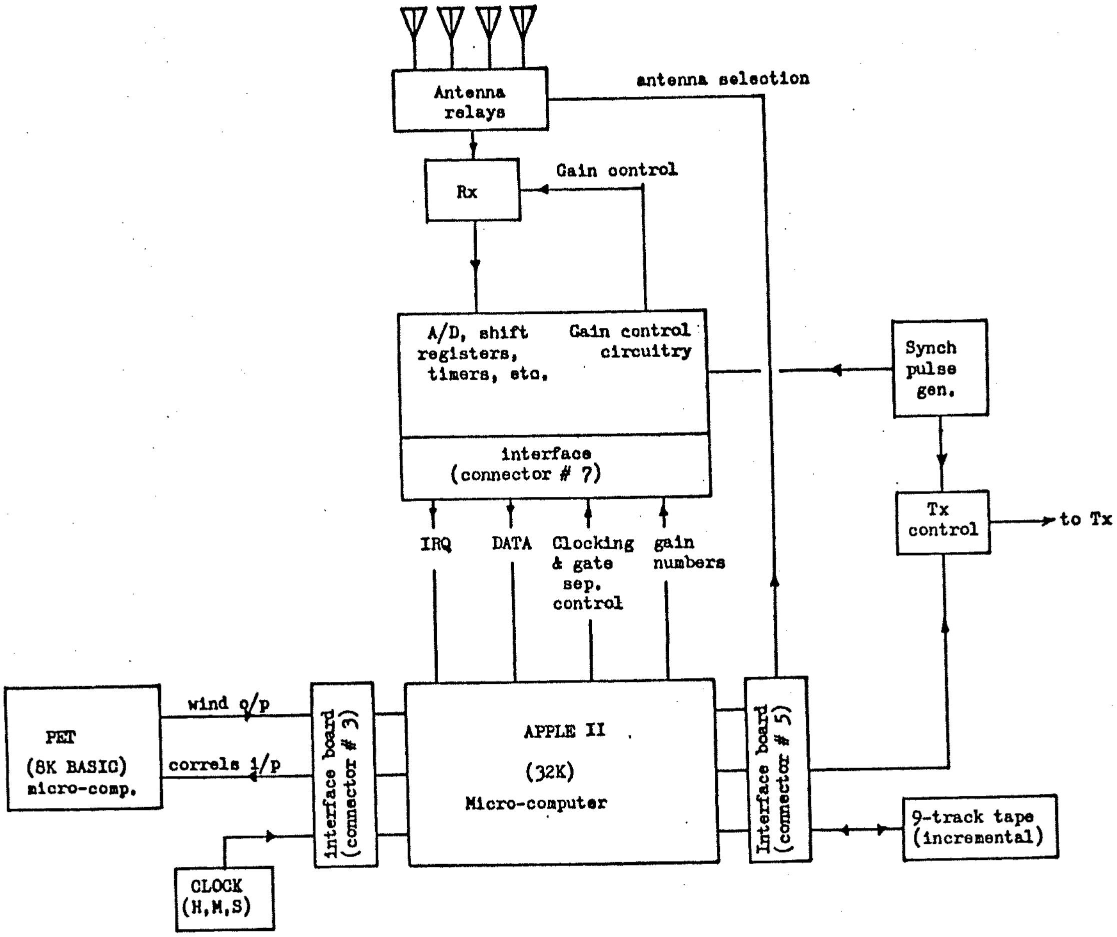

Antenna relays
antenna selection
Rx
Gain control
A/D, shift registers, timers, etc.
Gain control circuitry
Synch pulse gen.
interface (connector # 7)
Tx control
to Tx
IRQ
DATA
Clocking & gate sep. control
gain numbers
PET (8K BASIC) micro-comp.
wind o/p
correls i/p
Interface board (connector # 3)
APPLE II (32K) Micro-computer
Interface board (connector # 5)
9-track tape (incremental)
CLOCK (H,M,S)

Fig. 2

b) Interfaces to Apple computer

Three peripheral interface boards are used by the Apple II. These are the "analogue interface board" (pg. 14) which sets the receiver gain for each height gate, and samples the receiver output; the " tape interface board" (pg. 15) which controls the 9-track tape recorder and selects antenna relays through the relay board (pg. 16); and the interface board (not shown, but essentially the same as the tape interface board) connecting the Apple to the external clock and the Pet computer. These are plugged into the C700, C500, and C300 peripheral interface connectors respectively, on the Apple. The analogue interface is the most important, and will be described first.

(1) Analogue interface

The circuit diagram is shown on page 14 . The Tx synch pulse (which also interrupts the Apple through this interface) resets the logic and initializes a free running counter (IC7, pin 11) which puts out a 20 μsec square wave. This counter, like all others on the board, is driven by the Apple clock (Ø0, 1 MHz) or its sub-multiples. The Apple generates a software delay, when interrupted, (~50Km) and then loads a 3-bit binary number (variable GSEP) into the Johnson counter (IC9). This action enables the Johnson counter to start counting at the next positive transition of IC7-pin 11. It then puts out a string of evenly spaced 1 μsec pulses (the separation is determined by the 3-bit number; Table 2 shows available values). These pulses control both the A/D converter and the gain control circuitry. When a pulse is received by the A/D (IC16), it samples the receiver output (0-10V) through the sample and hold amplifier (IC15), converts it to binary (8 bits), and stores it in the 8x64-bit shift registers (IC1-4). When a pulse is received by the gain circuitry, it reads a 3-bit number from the 3x64-bit shift registers (IC18,19) which have been previously loaded by the Apple program, and uses this to select one of 8 preset voltages through a multiplexer (IC21). This voltage is used to control the gain of the external receiver. Thus the gain is being changed just as the receiver output is sampled; however, the receiver has a relatively long response time to a gain change, and so the output is not affected immediately.

Table 2.

Gate separation (μ sec)	loaded number (hex)
5	$2Ø
10	$1Ø
15	$3Ø
20	$Ø8
25	$28
30	$18
35	$38

As the amplitudes and gains are being clocked in under hardware control, the Apple program proveds a software delay (which has been determined in advance for the particular clocking rate used, and is defined by the variable CLKIN in the program) and terminates the clocking when exactly 64 amplitudes have been stored. The amplitude shift registers are now full, and the gain registers empty. The program then modifies the input to IC5 so that the amplitudes may be clocked out under program control, and later to IC20 so that the gain shift registers may be reloaded by the program for the next pulse. Figure 3 illustrates the sequence of events.

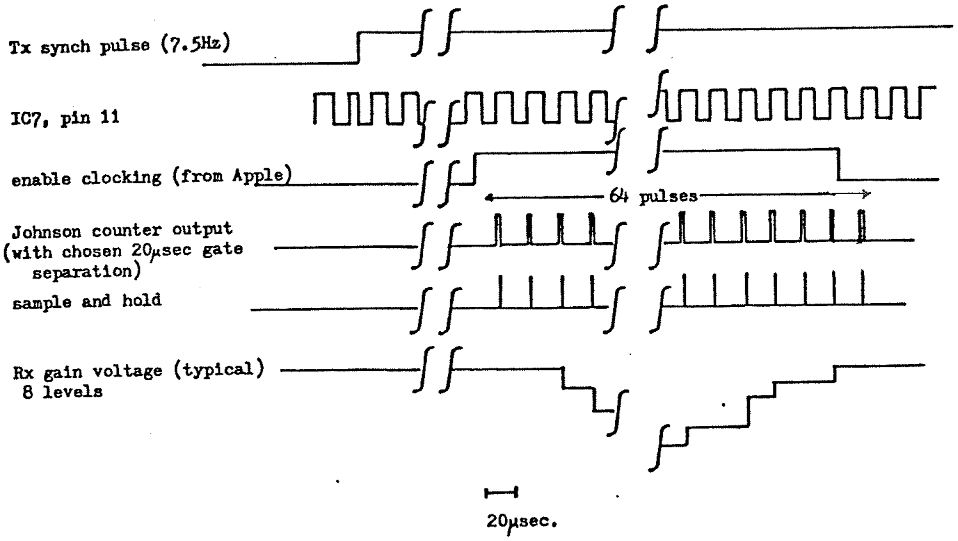

Tx synch pulse (7.5Hz)
IC7, pin 11
enable clocking (from Apple)
64 pulses
Johnson counter output (with chosen 20μsec gate separation)
sample and hold
Rx gain voltage (typical) 8 levels
20μsec.

Fig. 3

The reason for tying the sampling to the next positive transition of IC7-pin 11 after the Apple has sent the enable command is that some slack is permitted in the Apple software delay without affecting the actual delay of the first height gate from the transmitted pulse. This is necessary because the Apple must complete an instruction before responding to an interrupt request, which may take up to ~7μsec, depending on the type of instruction (in the present program the actual delay is ≲ 2μsec). (Note that the disable works in the same way)

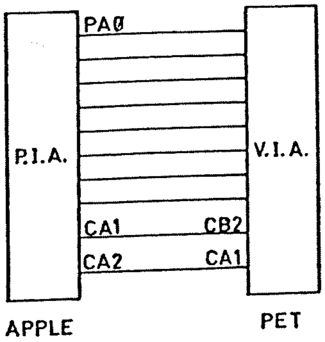
(2) Tape interface board

There are two peripheral interface adapters on this board; one is used for data and command output to the 9-track incremental tape recorder, and the other for tape signal lines (input), and antenna relay control (output). A spare line on the board is used for enabling the transmitter.

(3) Pet and clock interface board

Fig. 4 shows the connections between the P.I.A. (Apple) and V.I.A. (Pet).

Fig. 4

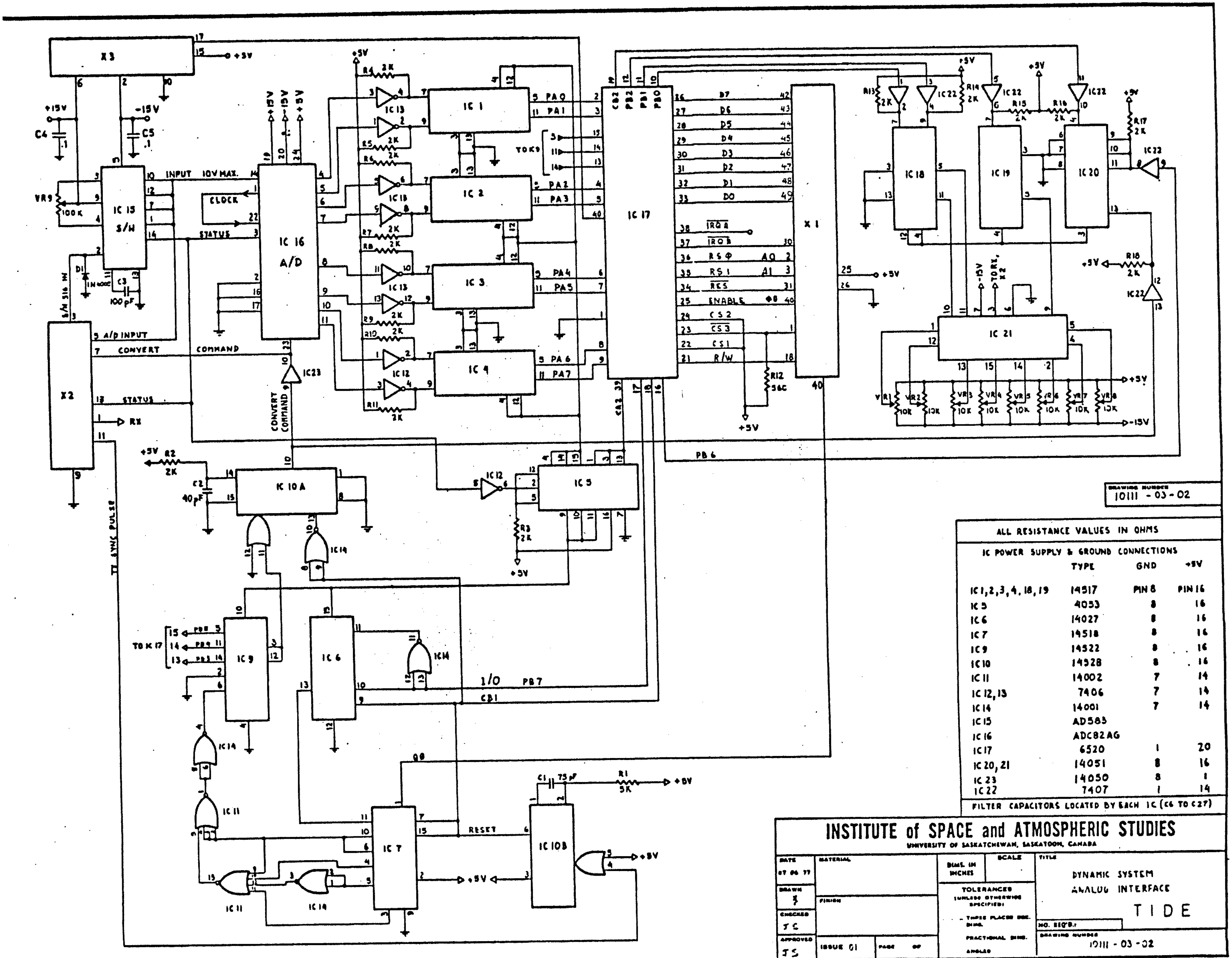


(c) Program operation

The program operation is as follows. After the program has been started it checks for a 0 or 5 min mark (seconds are ignored). When this is found it turns on the transmitter and waits to be interrupted by the synch pulse. The antennas are selected cyclically under program control. When a block has been stored, it is converted to binary between pulses - the mean signal being accumulated for use in setting gains in the next record. During the second and further blocks, correlations are also done between pulses. At the end of a record (which is determined by a block count) the average signal for the whole record is calculated, and the receiver gains set accordingly for the next record. At the beginning of the second record, the correlations calculated in the previous record are transferred to a second storage location from which they are fed to the Pet computer one height at a time. At the same time, new correlations are being calculated, and the results from the Pet calculation are received and stored on 9-track tape. All of these tasks are performed between the handling of incoming amplitudes.

Since the correlations are done as the record progresses, and only 6-7 seconds are required at the end to finish these, the record rate is not dependent on the Apple program. Calculation of wind values is rather time consuming however, and is is possible that the Pet may not finish in time for the next record. Consequently a new record is not initiated unless the Pet is finished with the last.

A comparison between winds from this real time system and a 'standard' system has been shown by Gregory et al. (1978).



DRAWING NUMBER 10111 - 03 - 02

ALL RESISTANCE VALUES IN OHMS

IC POWER SUPPLY & GROUND CONNECTIONS

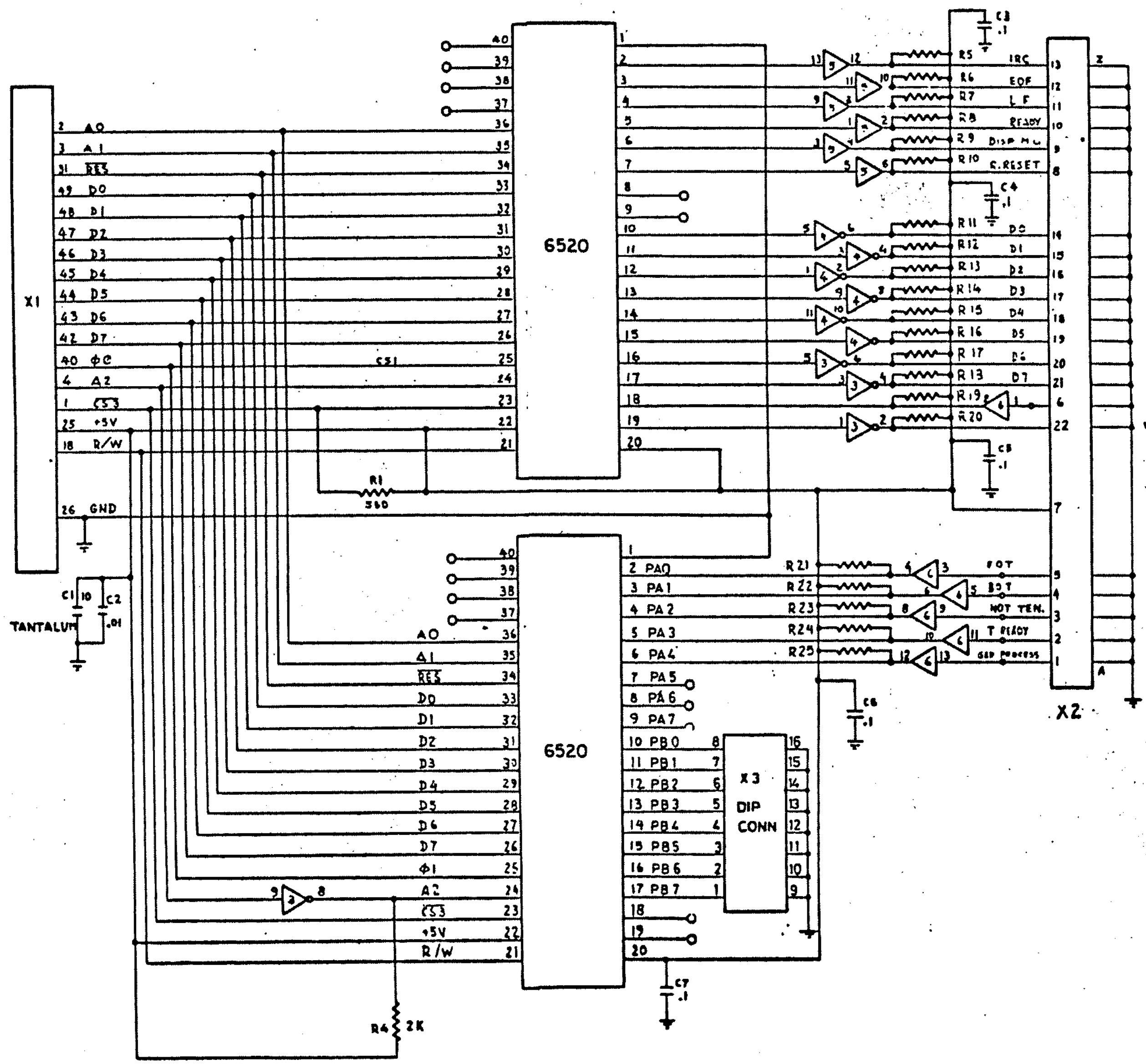
	TYPE	GND	+5V
IC 1,2,3,4,18,19	14517	PIN 8	PIN 16
IC 5	4053	8	16
IC 6	14027	8	16
IC 7	14518	8	16
IC 9	14522	8	16
IC 10	14528	8	16
IC 11	14002	7	14
IC 12, 13	7406	7	14
IC 14	14001	7	14
IC 15	AD583		
IC 16	ADC82AG		
IC 17	6520	1	20
IC 20, 21	14051	8	16
IC 23	14050	8	1
IC 22	7407	1	14

FILTER CAPACITORS LOCATED BY EACH IC (C6 TO C27)

INSTITUTE of SPACE and ATMOSPHERIC STUDIES

UNIVERSITY OF SASKATCHEWAN, SASKATOON, CANADA

DATE 07 06 77	MATERIAL	DIMS. IN INCHES	SCALE	TITLE DYNAMIC SYSTEM ANALOG INTERFACE TIDE
DRAWN Z	FINISH	TOLERANCES (UNLESS OTHERWISE SPECIFIED)		NO. REQ'D.
CHECKED JS		THREE PLACES DEC. DIMS.		
APPROVED JS	ISSUE 01 · PAGE OF	FRACTIONAL DIMS. ANGLES		DRAWING NUMBER 10111 - 03 - 02

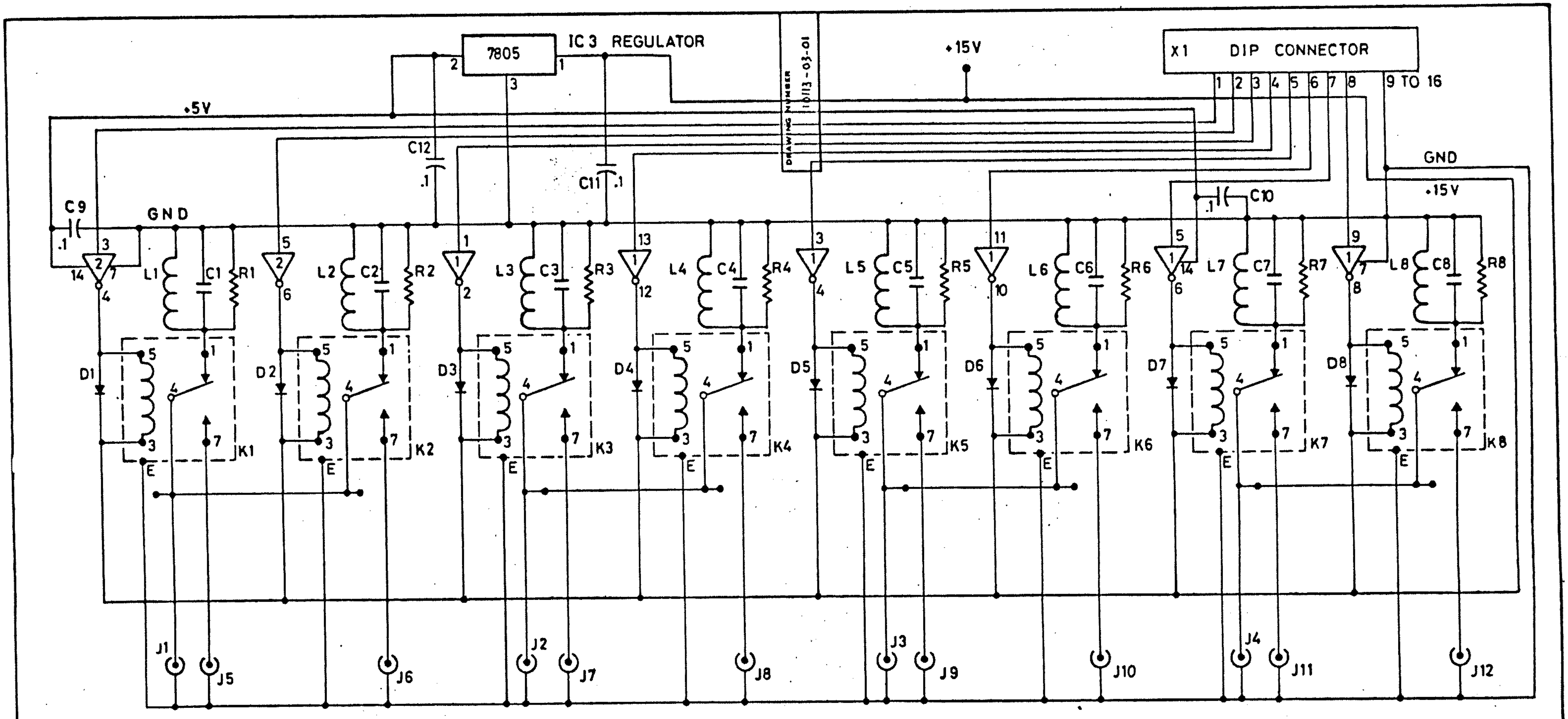


R5 THROUGH R20 = 180Ω EXCEPT
R19 = 2.2K OHMS
R21 TO R25 = 2.2K OHMS

DRAWING NUMBER
10112-03-01

INSTITUTE of SPACE and ATMOSPHERIC STUDIES
UNIVERSITY OF SASKATCHEWAN, SASKATOON, CANADA

DATE 20-07-77
DRAWN
CHECKED
APPROVED
MATERIAL
FINISH
ISSUE
PAGE OF
DIMS. IN INCHES
SCALE
TOLERANCES (UNLESS OTHERWISE SPECIFIED)
THREE PLACED DEC. DIMS.
FRACTIONAL DIMS.
ANGLES
TITLE
TAPE INTERFACE
NO. REQ'D.
DRAWING NUMBER
10112-03-01



RELAYS SHOWN IN NON-ENERGIZED MODE

DRIVER DEVICES = 7406

ALL DIODES 1N4007

ALL RESISTANCE VALUES IN OHMS

ALL CAPACITANCE VALUES IN PICOFARADS

J1 THRU J12 BNC CONNECTORS

INSTITUTE of SPACE and ATMOSPHERIC STUDIES

UNIVERSITY OF SASKATCHEWAN, SASKATOON, CANADA

DATE 10-08-77

DRAWN ⁊

CHECKED

APPROVED

MATERIAL

FINISH

ISSUE

PAGE OF

DIMS. IN INCHES

SCALE

TOLERANCES (UNLESS OTHERWISE SPECIFIED)

.........THREE PLACES DEC. DIMS.

. .... FRACTIONAL DIMS.

. ... ANGLES

TITLE

RELAY BOARD

TIDE

No. Req'd.

DRAWING NUMBER 10113-03-01

IV. Operating problems

This chapter outlines some of the problems met in practice and their resolution, or ways in which they may be overcome in future modifications.

a) Receiver gain control

With the present receiver it is not possible to change the gain in 20μsec (the usual gate separation) because of the narrow bandwidth. Some re-design is proceeding which will place the gain control after the I.F. stage, but this will probably remain a problem. In the program (Version 17d,Apple) the attempt to set a different gain for each gate depending on the measured mean amplitude failed because the gain could not be set independently for each gate. The result was that the gain (and signal) usually 'locked' into a herringbone pattern, since the gain for one height gate had more effect on the next. A partial solution (version 18, Apple) is to set the gain for groups of four adjacent height gates on the basis of the maximum signal in the last three, since the gain for the first is uncertain. This allows overloading in the first gate of every four, since its signal is not considered, however it has turned out to be only a minor problem.

b) Overloading

Appendix B, section 5 (pg.63) describes the serious effect of overloading on the correlation values. It is not possible to avoid overloading completely because of unpredictable changes in signal strength occurring over periods longer than a record length. This problem can probably be resolved by modifying the Apple program to save separate means of the binary sequences (i.e. separate zero-lag auto correlations) instead of the present practice of using an 'average' over all (three) antennas for the calculation of cross correlations in the Pet.

c) A/D converter

The particular A/D converter used does not tolerate overloading. For minor overloading ($\gtrsim$ 10V) the output is zero; for greater overloading it is unpredictable-thus the substitution of $FF for $ØØ in the Apple program. A final solution was to put a 9.9V zener diode across the receiver output.

d) Pet

It was found that the temporary data storage (1 page, see Appendix C,pg. 78) was not safe from the Basic system even though the internally calculated storage requirements seem to leave enough room, and no string operations are used. Consequently, the transfer of all data into Basic program variables was made as soon as possible in the program. It would be more efficient to check the auto correlation for fast fading first and only read the rest if necessary, since the 'PEEKING' of data out of storage is time consuming. However, the present program does finish a record in less than the required time (5 min) so nothing will be done about this at present.

V. Conclusions and future work

The real time system has been operating since Sept. 1978 with the gain control modification described in Chapter 4. The amount of output is ∽ 10-20% less than for the previous system in which raw data (amplitudes) were recorded and analysed at great expense on a large computer, presumably due to the binary method in use; but the quality of the resulting data is essentially the same. Some increase in output is expected when separate means are used in the calculation of cross correlations.

Since incremental tape recorders are becoming obsolete, a future modification of the program will be made to save the output for a full record and dump it onto tape at the end of the record. The present four antenna array is unique, thus a modification will be made to allow operation on the more usual three antenna array (commonly an equilateral or right angle triangle arrangement). A slightly lower pulse rate (∽6Hz instead of 7.5) and longer block length (80 pulses/antenna instead of the present 64) will be employed.

REFERENCES

Briggs, B.H. 1967 "Instruction manual for drifts", ADP42, Dept. of Physics, Univ. of Adelaide

Fedor, L.S. 1967 J.G.R. 72, 5401

Gregory, J.B., A.H. Manson, D.G. Stephenson, C.E. Meek 1978 "A Spaced-antenna Winds System using Microprocessors", Institute of Space and Atmospheric Studies, Univ. of Sask., Saskatoon, Canada.

Hald, A. 1952 "Statistical Theory with Engineering Applications", Wiley (New York)

Hoel, P.G., S.C. Port, C.J. Stone 1971 "Introduction to Statistical Theory", Houghton-Mifflin Co. (Boston)

Meek, C.E. 1978 "The Analysis of Ionospheric Drift Measurements", Institute of Space and Atmospheric Studies, Univ. of Sask. Saskatoon, Canada

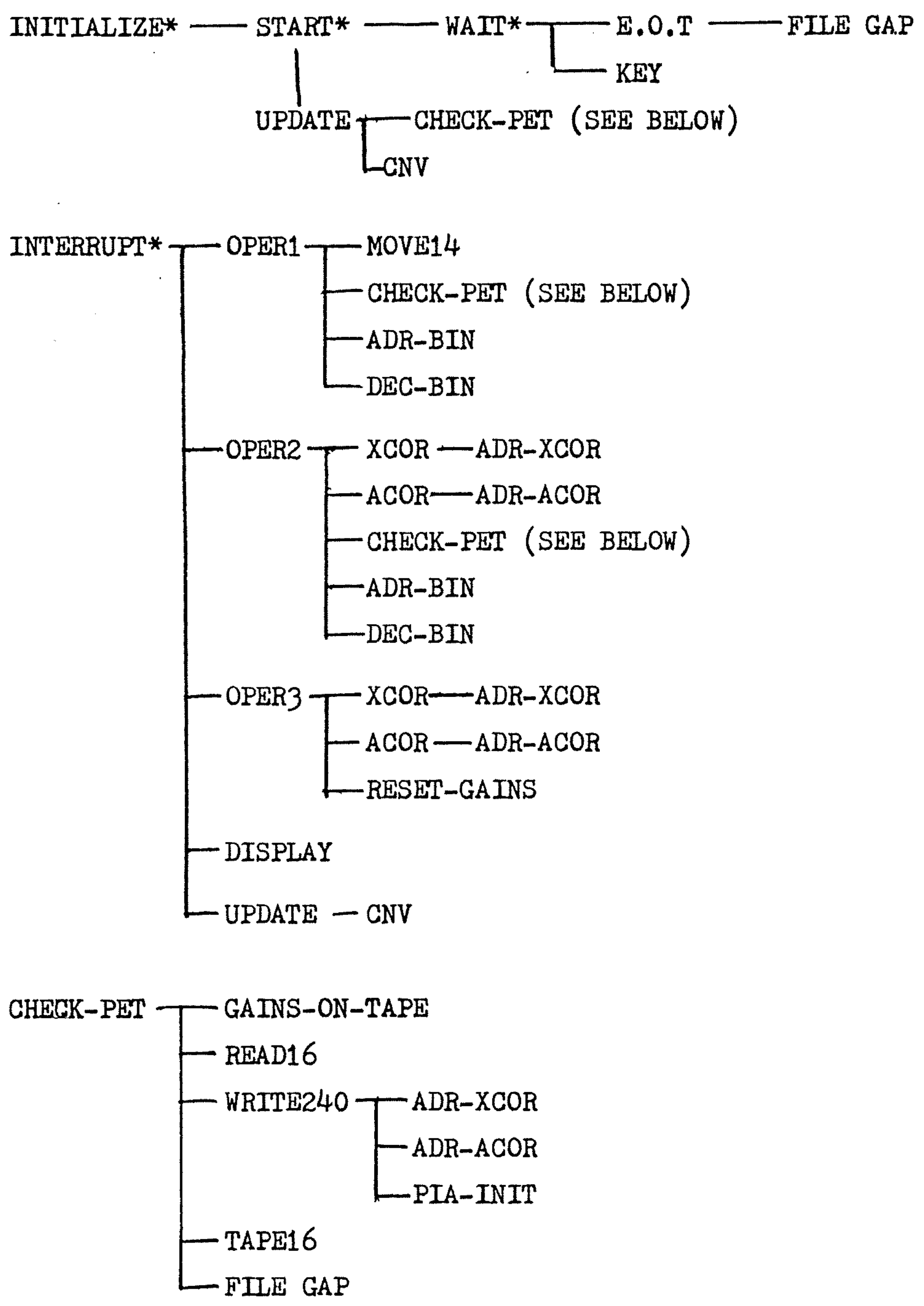
Phillips, G.J., M. Spencer 1955 Proc. Phys. Soc. , B68, 481

APPENDIX A

Apple program details

1. Subroutine calling diagram
2. Detailed description of routines
3. Method of calculating correlation
4. Some flow charts
5. Apple bulk storage map
6. Volatile zero page storage
7. Zero page storage (constants, control flags, counters)
8. Program listing (version 17d, 4 antennas)

1. SUBROUTINE CALLING DIAGRAM (4 ANTENNAS)
(* - NOT A SUBROUTINE)



2. Short description of routines

START : Updates clock store and checks for keyboard commands between records; checks for start time (minute units = 0,5) if the Pet is finished with the previous record. It initiates a record by setting a flag and enabling the interrupt, and then sits in a short-instruction loop waiting to be interrupted by the synch pulse. **At the end of a record it disables the interrupt.**

INITIALIZATION : Initializes the gain settings for the 32 height gates (low to high) to 7,7,7,7,6,6,6,6, ... 0,0,0,0. Reads in day#, and # blocks per record; sets bottom delay, gate separation, and clockin delay parameters; initializes the P.I.A.'s; chooses the first antenna relay; runs the gain settings into the P.I.A., and sets cycling parameters and other flags.

UPDATE : Reads H,M,S from external clock and stores in clock location and on monitor; checks for day# change; sets file gap flag when day# is changed; services the Pet (and Tape recorder) through CHECK-PET routine if a record is not in progress.

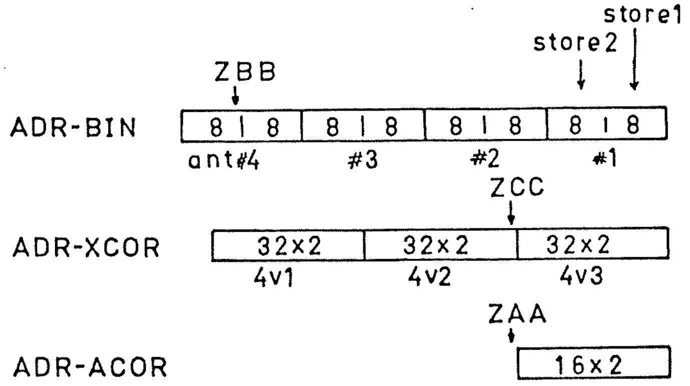
INTERRUPT : Saves accumulator and X register (not needed) on stack; gives software delay before first height gate; turns on clocking, waits for 64 gates, stops clocking; runs in gains for next pulse. If a record is not in progress (this is a faint possibility since the system could come out of interrupt at the end of a record and be interrupted again before the interrupt is disabled) it returns from interrupt at this point. It reads 32 amplitudes from the P.I.A., loads them into block storage and selects the next antenna relay. If the first block of the record is in progress, then OPER1 is called; if any other block, then OPER2; if the record has ended (by block count) then OPER3 is called.

OPER1 : Operations in this routine depend on the pulse count (0-$FF). After the first three pulses it shifts the auto, cross and date/time store 1 to store 2 (for output to Pet) and zeroes store 1; stores the present date/time in store 1 and zeroes the binary storage (store 1 and 2). After all other pulses, except the last four, it services the Pet and tape recorder (through CHECK-PET). As the block is finished for each antenna, the data are converted to binary through DEC-BIN.

OPER2 : Again the operations depend on the pulse number. After the first 96 pulses it cycles around the cross correlation (XCOR), auto correlation (ACOR), and CHECK-PET routines. The idea is to service the Pet as often as possible since its calculations constitute the limit on the record rate. After all other pulses, except the last four, it calls CHECK-PET. At the end of the block (last four pulses) the data are converted to binary.

OPER3 : This is called after the last block in the record has been converted to binary, and it runs through the cross and auto correlation routines twice to finish up. Then RESET-GAINS is called to calculate the average signal and choose the gain settings for the next record. The record is terminated in this routine by resetting REC.

ADR-BIN, ADR-XCOR, ADR-ACOR : These calculate the address to the appropriate binary sequences and the accumulators for store 1 cross and store 1 auto correlations from the input height number (0-$1F) as shown below:



Note that the address points to the byte below the 'boundary' for programming reasons.

DEC-BIN : Averages signal ($\div 64$) for a given antenna number (0-3) at 32 heights, and adds this to the $\Sigma\,\overline{sig}$ store (for future use in setting gains and for output to tape): avge = Σsig/64 + 1 (truncated). This average is compared with each amplitude in the block to get the binary representation of the sequences (amplitude $\geq$ avge gives a '1' bit) which is put into store 2 binary (see Apple bulk storage map, pg. 35). This process takes ~ 0.08 sec for one antenna at 32 heights.

XCOR : Does 'partial' cross correlations (4v1, 4v2, 4v3) for one height. The linear (instead of loop) section to add 1's was found to be necessary for saving time. The method is described in detail in Section 3 (pg.27). Required time is ~ 0.06 sec.

ACOR : Does 'partial' auto correlation for one height (0 to 15 lags) over antennas #1,2,3 only (see Section 3, pg. 28). Required time is ~ 0.06 sec.

RESET-GAINS : The first half of this routine divides the $\Sigma'\,\overline{sig}$ by the (number of blocks times the number of antennas) to get the average signal, $\overline{sig}$, for the whole record; and stores this and the present gain settings for output to tape. The second half compares $\overline{sig}$ with 80 and 160 ($5Ø and $AØ) at each height. If it is between these limits the gain is left as is at that particular height; if greater then the gain is reduced by one step (if possible); if less then the gain is increased by one step (if possible). Ideally each gain step (adjusted manually) should be equal to the ratio of the two limits used (e.g. 80, 160), which is 6 dB bere, otherwise the gain will sometimes bounce back and forth from too high to too low, even when the average signal strength is constant. The upper limit is restricted by overloading: overloading usually begins when the mean reaches ~140, and may be 5 or 10% at a mean of 160. At present, gain steps of ~8dB are used since 48 dB (=6 dB x 8 gain steps) may not quite cover the range of signals encountered in practice.

CHECK-PET : This routine controls the tape recorder and services the Pet computer. The latter is programmed to output 16 bytes and input 240 bytes each time it is accessed. Operations are governed by flags best described by the flow chart (pg. 34). Tasks are made short enough to fit between pulses (except for the file gap ~ but this is put on only once per day after the fourth pulse in a record). Note that at the end of a record the Pet is holding the last output and waiting for the next input, which means that the 'true' end of the record as far as the CHECK-PET routine is concerned, is during the beginning of the next.

TAPE16 : Puts 16 bytes on tape (1.2 msec/byte with the present tape recorder) from the temporary Pet output store.

WRITE240 : Feeds 240 bytes to Pet (date/time, cross and auto for one height).

READ16 : Takes 16 bytes from Pet and puts them is a temporary store.

FILE-GAP : Puts a file gap on tape and waits until it is finished (0.5 sec). It also takes the present day # and puts it back into the manually initialized store - this is just a safety feature in case someone resets the Apple and forgets to re-initialize the day#.

DISPLAY : Displays a character corresponding to the ASCII graphic for the lowest 4 bits for 17 bytes (designated by a pointer to one byte below the desired bytes, which is initialized manually before running the program).

0-9	come	out	as	0-9
A	comes	out	as	:
B	"	"	"	;
C	"	"	"	<
D	"	"	"	=
E	"	"	"	>
F	"	"	"	?

3. Method of continuous correlation calculation (4 antennas)

Two consecutive blocks of data (n, n+1) are available in binary form as shown in Fig. 1 (at the beginning and end of the calculation, one of then will be all zeroes). Antenna #4 is correlated with the other three at each lag (because of the special antenna array used)- so just one pair will be described. Ten bytes of antenna #4 are transferred to zero page as shown in Fig. 2. This situation will be arbitrarily called a negative lag, and since #4 is shifted 2 bytes (16 bits) to the right, this is lag -16. An 'AND' operation is performed between the 8 pairs of bytes indicated - the result of an 'AND' contains a '1' bit if the corresponding bits in the pair of bytes 'ANDed' were both 1's, and is a 'Ø' bit otherwise. The number of '1' bits in each result is accumulated for the 8 ANDs and is added to the 4v3 lag -16 accumulator (2 bytes in store 1 cross correlations -see Apple bulk storage map, pg. 35) The zero page antenna #4 sequence (12 bytes, although only 10 used) is now shifted left by 1 bit using the 'ROL' instruction. A zero bit is inserted on the RHS for each set of rolls, so zeroes fill the space left by the shifting sequence. Now the same 8 bytes of antenna #3 are ANDed again for lag -15. Note that the leftmost bit of the antenna #4 sequence is now outside the boundary of the 8 bytes, and does not come into the AND results here.

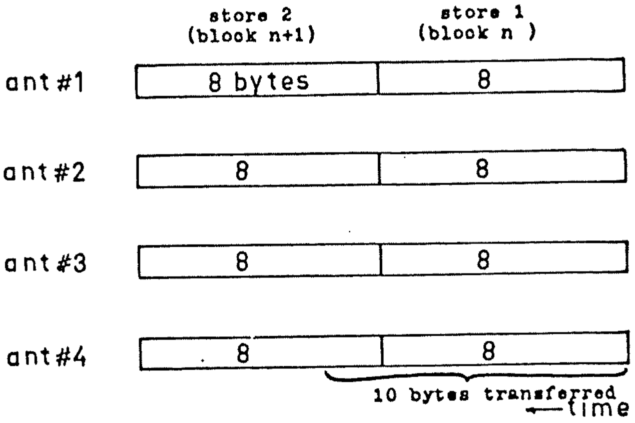


Fig. 1

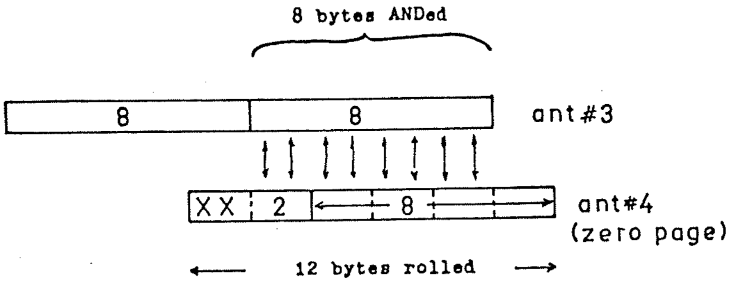


Fig. 2

The procedure continues down to lag -1. Before lag 0 (when antenna #3 and #4 are perfectly aligned), the two extra bytes in antenna #4 (LHS) are zeroed. Now ten bytes are ANDed as shown in Fig. 3 (12 bytes are still rolled for easier programming). For zero and +ve lags, no bytes of antenna #4 will be outside the AND operation, but some of the antenna #3 bits effectively will be since they are being ANDed with zero bits.

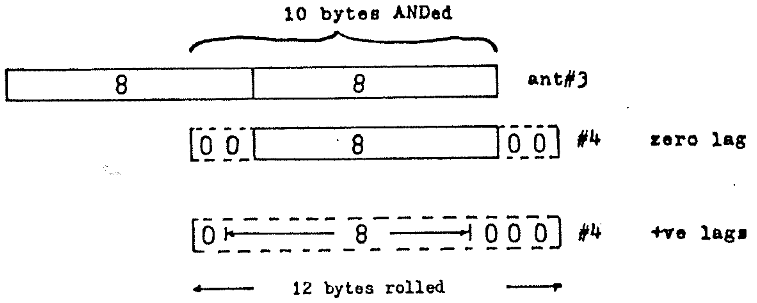
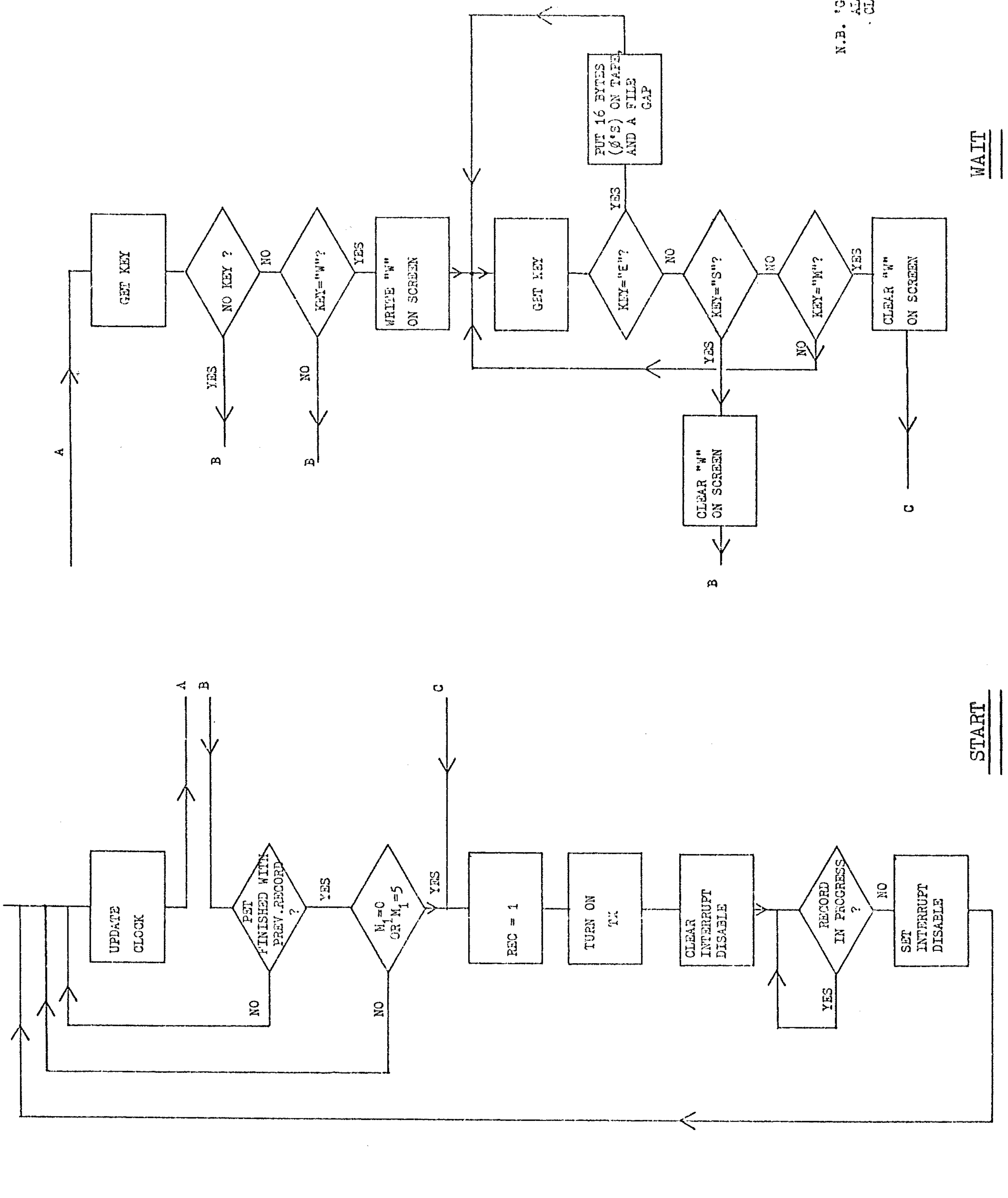


Fig. 3

The auto correlation is almost exactly the same except that only antennas # 1,2,3 are used, and just zero and +ve lags are calculated (the sequences are ANDed with themselves); the results for each lag being added into the same accumulator - effectively giving an 'average' auto correlation at each lag .

At the end, store 2 binary is moved to store 1 and store 2 is zeroed (the latter is actually only required at the end of the record). When the next block has been converted to binary, it is placed in store 2, and the same correlation procedure is followed again. The final result of this process is exactly the same as if the binary sequences for the whole record were correlated at once. The accumulators (auto and cross) contain the number of 1-matches for each lag. The zero lag auto correlation contains the total number of 1's in the three (antenna #1,2,3) sequences - this is used for calculating the actual correlation value from the number of 1-matches (done in the Pet) as described in Appendix B, Section 1 (pg. 57).

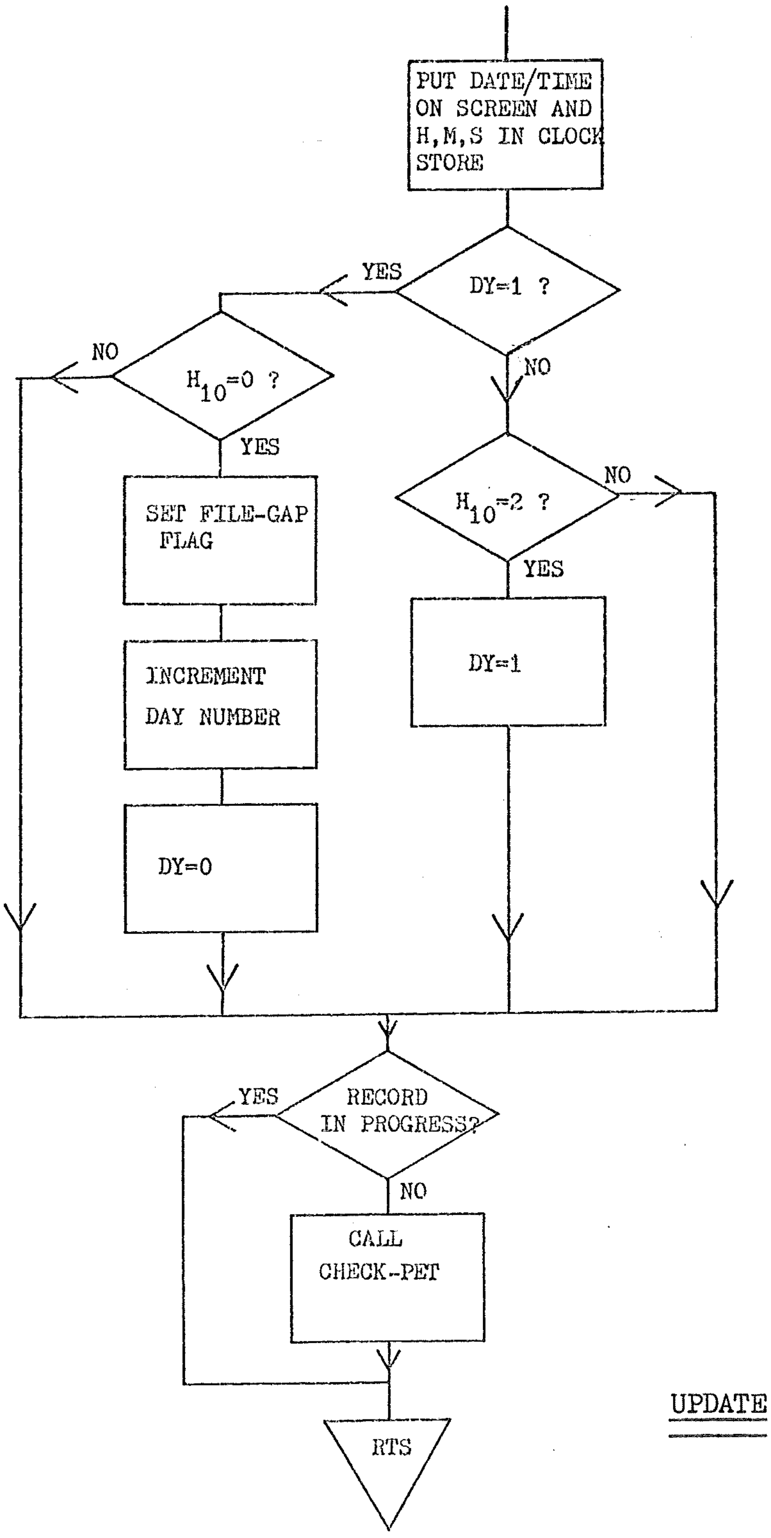
4. Flow charts



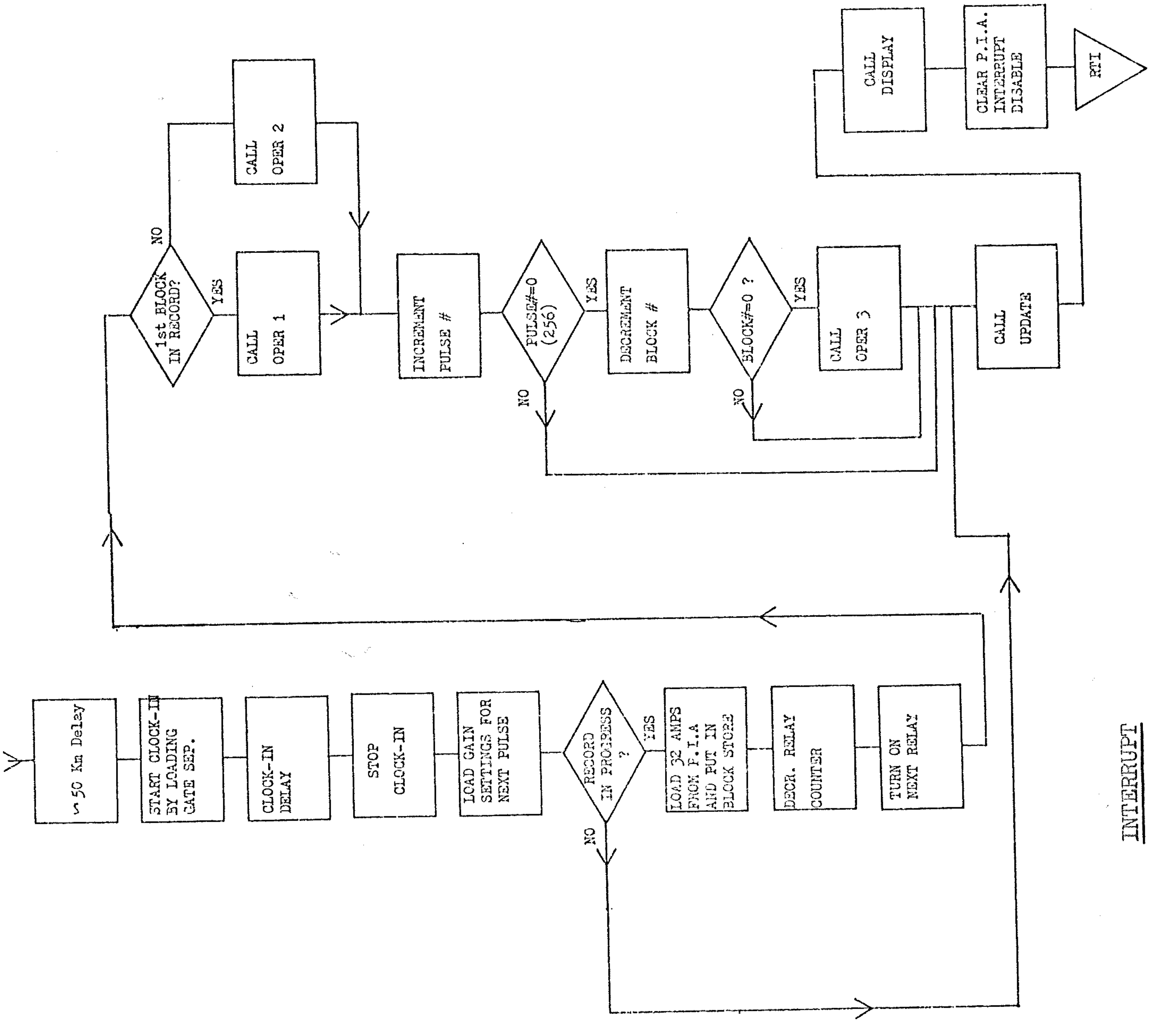
WAIT

START

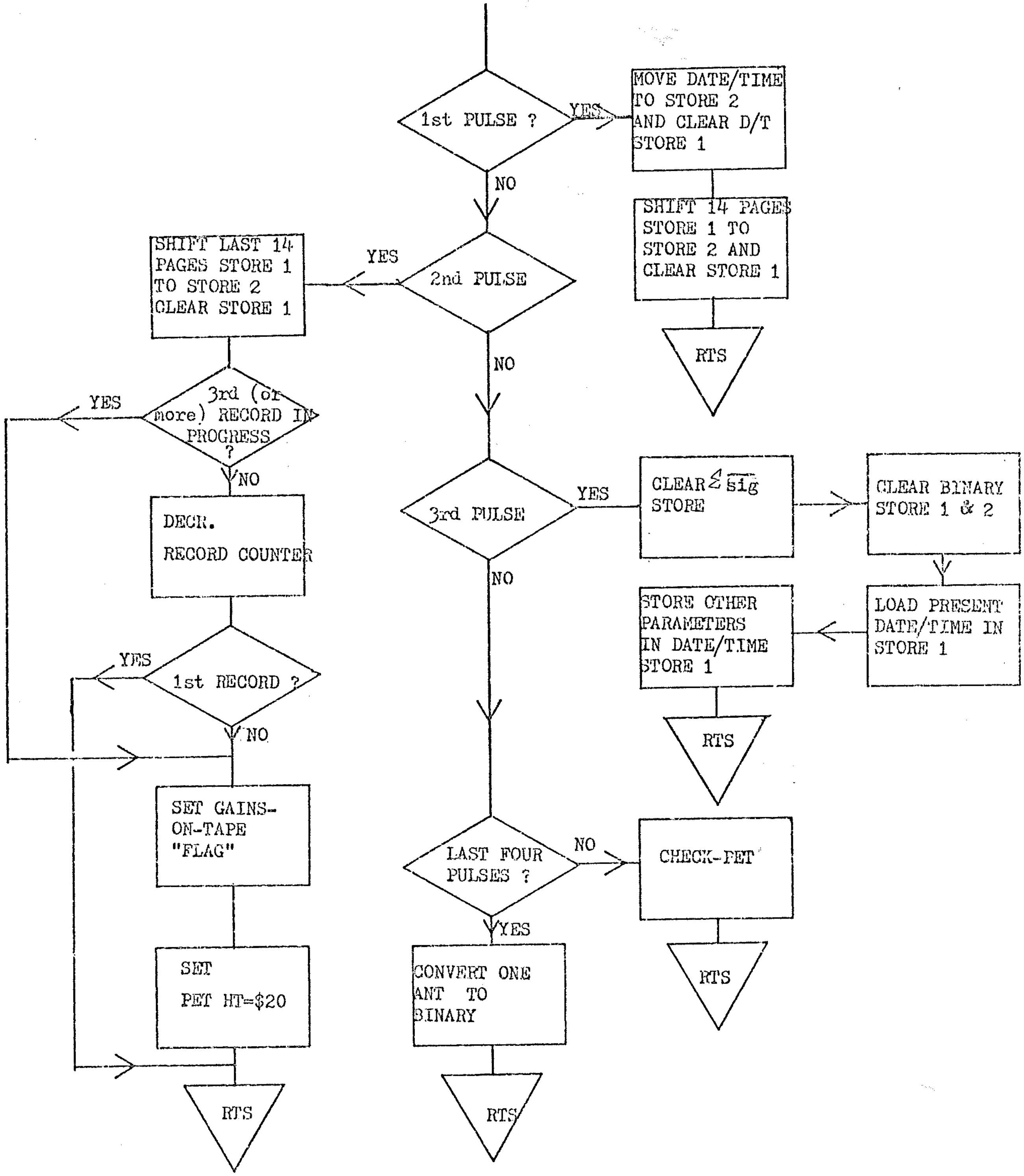
N.B. 'GET KEY' ROUTINE ALSO UPDATES THE CLOCK


PUT DATE/TIME ON SCREEN AND H,M,S IN CLOCK STORE
DY=1 ?
YES
NO
H_{10}=0 ?
NO
YES
SET FILE-GAP FLAG
INCREMENT DAY NUMBER
DY=0
H_{10}=2 ?
NO
YES
DY=1
RECORD IN PROGRESS?
YES
NO
CALL CHECK-PET
RTS

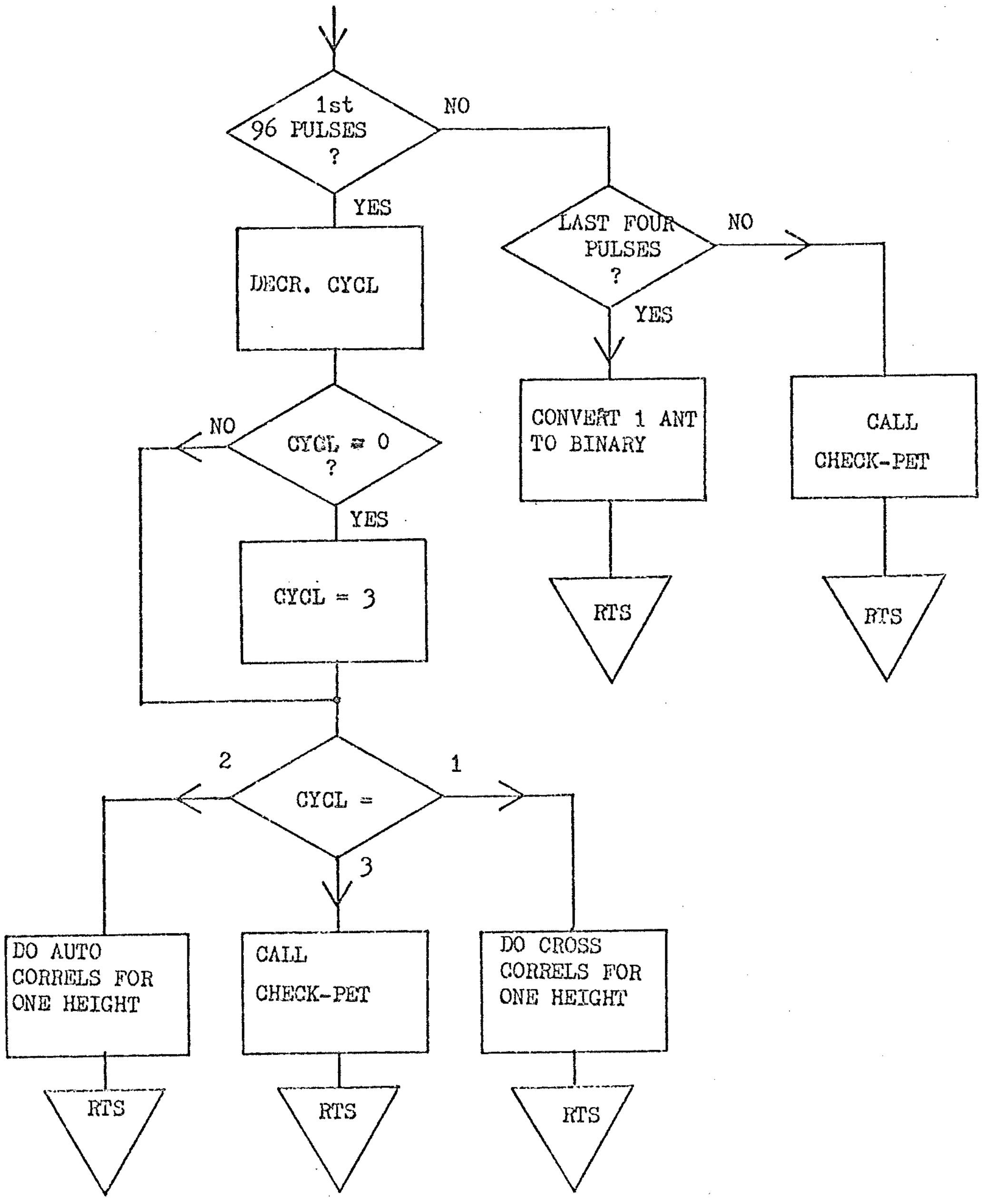
UPDATE


~50 Kn Delay
START CLOCK-IN BY LOADING GATE SEP.
CLOCK-IN DELAY
STOP CLOCK-IN
LOAD GAIN SETTINGS FOR NEXT PULSE
RECORD IN PROGRESS ?
YES
NO
LOAD 32 AMPS FROM P.I.A AND PUT IN BLOCK STORE
DECR. RELAY COUNTER
TURN ON NEXT RELAY
1st BLOCK IN RECORD?
YES
NO
CALL OPER 1
CALL OPER 2
INCREMENT PULSE #
PULSE#=0 (256)
YES
NO
DECREMENT BLOCK #
BLOCK#=0 ?
YES
NO
CALL OPER 3
CALL UPDATE
CALL DISPLAY
CLEAR P.I.A. INTERRUPT DISABLE
RTI

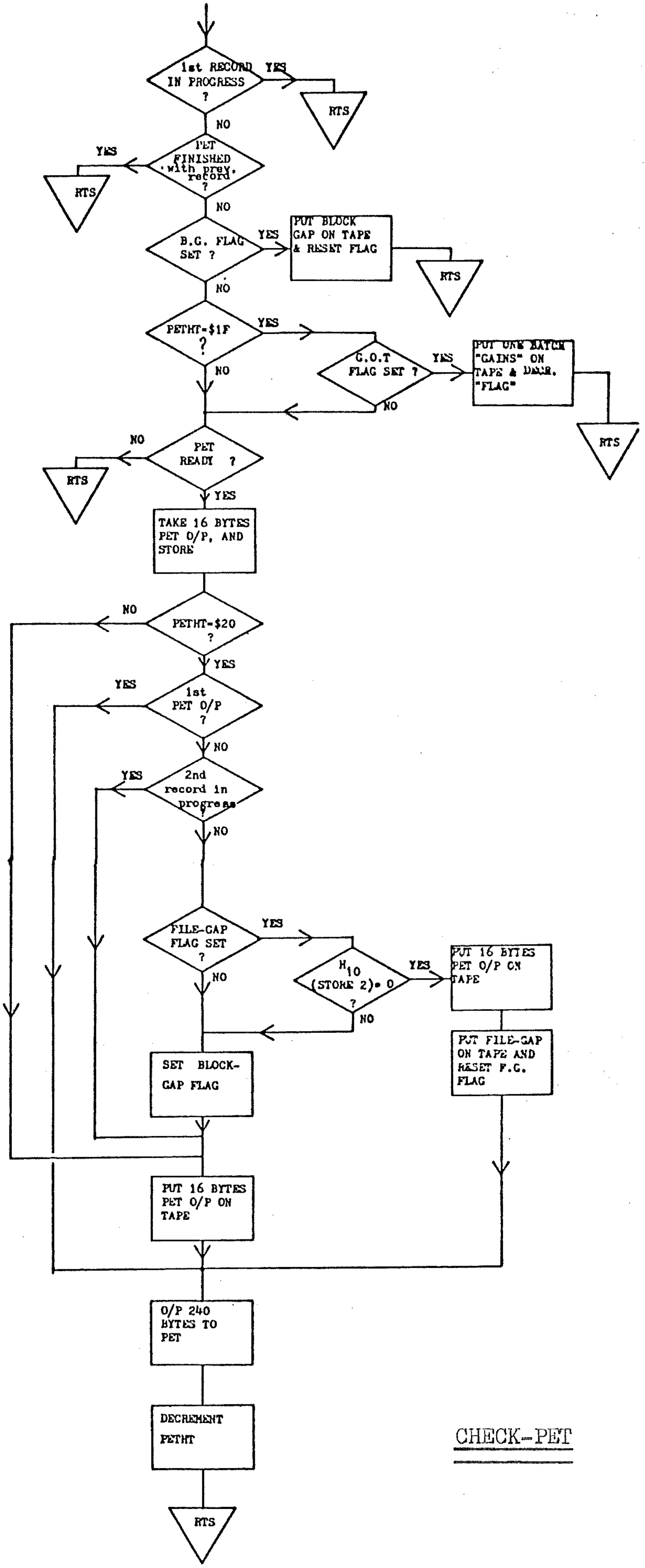
INTERRUPT


1st PULSE ?
YES
MOVE DATE/TIME TO STORE 2 AND CLEAR D/T STORE 1
SHIFT 14 PAGES STORE 1 TO STORE 2 AND CLEAR STORE 1
RTS
NO
2nd PULSE
YES
SHIFT LAST 14 PAGES STORE 1 TO STORE 2 CLEAR STORE 1
3rd (or more) RECORD IN PROGRESS ?
YES
NO
DECR. RECORD COUNTER
1st RECORD ?
YES
NO
SET GAINS-ON-TAPE "FLAG"
SET PET HT=$20
RTS
NO
3rd PULSE
YES
CLEAR Σ sig STORE
CLEAR BINARY STORE 1 & 2
LOAD PRESENT DATE/TIME IN STORE 1
STORE OTHER PARAMETERS IN DATE/TIME STORE 1
RTS
NO
LAST FOUR PULSES ?
NO
CHECK-PET
RTS
YES
CONVERT ONE ANT TO BINARY
RTS

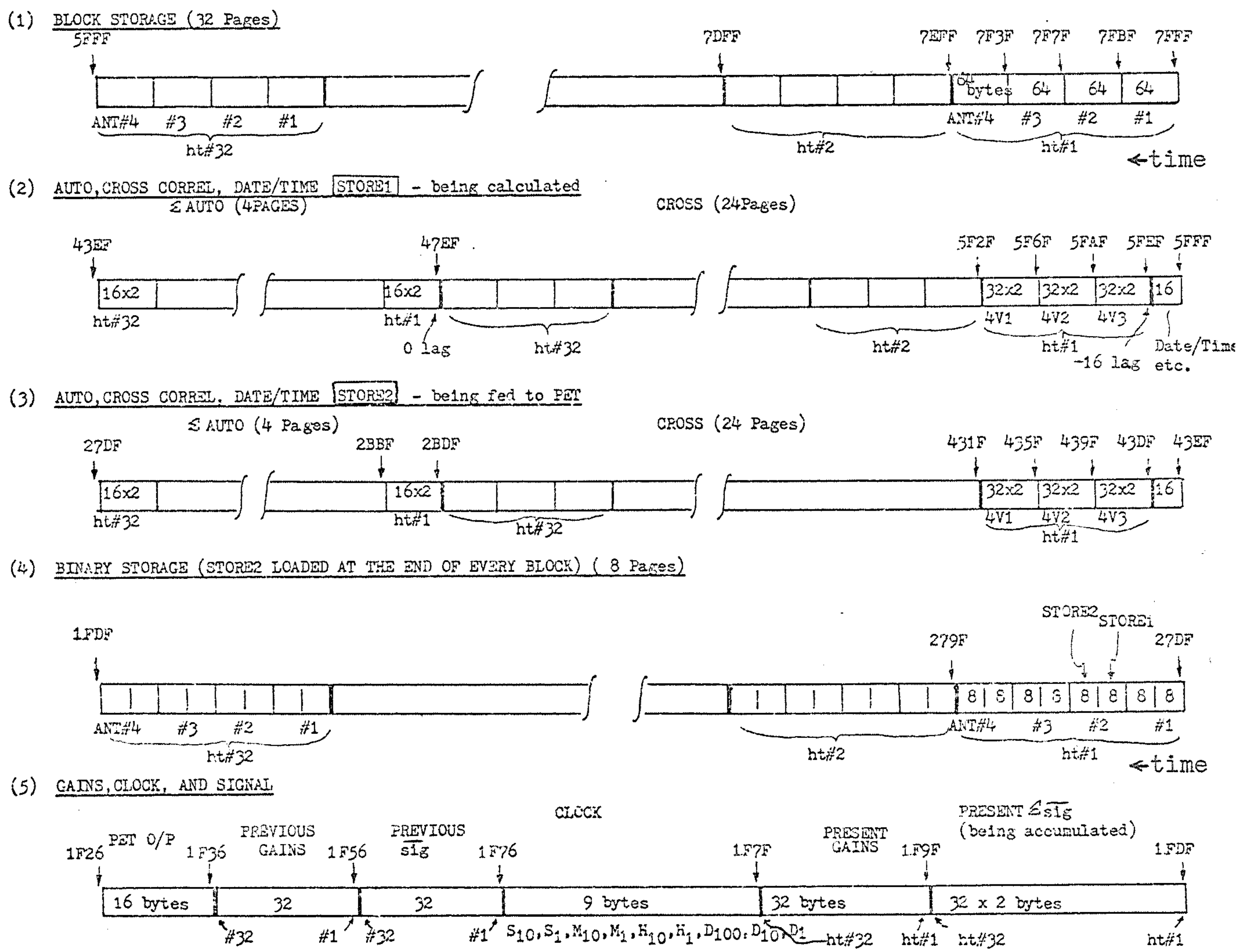
OPER 1


1st
96 PULSES
?
NO
YES
DECR. CYCL
LAST FOUR
PULSES
?
NO
YES
CONVERT 1 ANT
TO BINARY
CALL
CHECK-PET
RTS
RTS
NO
CYCL = 0
?
YES
CYCL = 3
2
CYCL =
1
3
DO AUTO
CORRELS FOR
ONE HEIGHT
CALL
CHECK-PET
DO CROSS
CORRELS FOR
ONE HEIGHT
RTS
RTS
RTS

OPER 2


1st RECORD IN PROGRESS ?
YES
RTS
NO
PET FINISHED with prev. record ?
YES
RTS
NO
B.G. FLAG SET ?
YES
PUT BLOCK GAP ON TAPE & RESET FLAG
RTS
NO
PETHT=$1F ?
YES
G.O.T FLAG SET ?
YES
PUT ONE BATCH "GAINS" ON TAPE & DECR. "FLAG"
RTS
NO
NO
PET READY ?
NO
RTS
YES
TAKE 16 BYTES PET O/P, AND STORE
PETHT=$20 ?
NO
YES
1st PET O/P ?
YES
NO
2nd record in progress ?
YES
NO
FILE-GAP FLAG SET ?
YES
H_{10} (STORE 2)= 0 ?
YES
PUT 16 BYTES PET O/P ON TAPE
PUT FILE-GAP ON TAPE AND RESET F.G. FLAG
NO
NO
SET BLOCK-GAP FLAG
PUT 16 BYTES PET O/P ON TAPE
O/P 240 BYTES TO PET
DECREMENT PETHT
RTS

CHECK-PET



5. APPLE BULK STORAGE (the low order byte in two byte accumulators has the higher address)

	ROUTINE												
LOCATION	OPER1	MOVE 14	OPER2	OPER3	ADR-BIN	ADR-XCOR	ADR-ACOR	XCOR	ACOR	DEC-BIN	CHECK-PET	RESET-GAIN	INTER-RUPT
9E													
9F								ZNUM					
Aø	ZANT		ZANT					R		ZANT		ZQUO	
A1	ZD	ZD	ZHT	ZHT	ZHT	ZHT	ZHT	O	R	ZDD	ZHT	ZD	
A2	+ 1	+ 1			ZTEMP	ZTEMP	ZTEMP	L	O	+ 1	ZD	+ 1	
A3	ZE	ZE						L	L	ZHT	+ 1	ZHT1	
A4	+ 1	+ 1						I	L	ZBIN		ZTEMP	
A5								N	I	ZBYTE			AD
A6								G	N	ZBYTE1			+ 1
A7									G	ZAVG			
A8													
A9													
AA							ZAA		ZAA		ZAA		
AB							+ 1		+ 1		+ 1		
AC					ZBB			ZBB	ZBB	ZBB			
AD					+ 1			+ 1	+ 1	+ 1			
AE						ZCC		ZCC	ZBB1		ZCC		
AF						+ 1		+ 1	+ 1		+ 1		
Bø								ZBZ	ZBZ				
B1								+ 1	+ 1				
B2								ZCC1	ZBB2				
B3								+ 1	+ 1				
B4								ZBB1	ZANT				
B5								+ 1	ZLAG				
B6								ZXC	ZSUM				
B7								ZLAG					

6. VOLATILE ZERO-PAGE STORAGE FOR SUBROUTINES

7. ZERO PAGE STORAGE

(CONSTANTS*, CONTROL FLAGS, COUNTERS)

LOCATION	NAME	USE	ROUTINE
ØØB8	DELAY*	Controls delay before 1st ht. gate	INTERRUPT
B9	GSEP*	Determines gate separation (20μs at present)	"
BA	CLOCKIN*	Controls delay for clocking in amplitudes must cover 1st 64 ht. gates exactly	"
BB	REC	= 1 if record in progress	VARIOUS
BC	DY	= 1 if looking for day # update	UPDATE
BD	NBLK	= NBLK1 at record start, decremented to zero, counts # blocks taken	interrupt
BE	NBLK1*	# BLOCKS/RECORD -chosen at system start	"
BF	PLSE	PULSE count (0-255) for each block, also used to calculate addresses, and select operations in OPER1, OPER2	INTERRUPT OPER1, OPER2
CØ	NRELAY	decremented with every pulse, used to choose relay #, also used to calculate addresses.	INTERRUPT
C1	NRELAY1*	= 4 (# ANTENNAS USED), used to reload NRELAY	INTERRUPT
C2	PET HT	= $20 when STORE2(AUTO & CROSS) filled, decremented to zero as PET processes these.	CHECK-PET
C3	B.G. FLAG	= 1 if block gap required	CHECK-PET
C4	RECORD#	indicates whether 1st or 2nd record in progress	CHECK-PET & others
C5	1stPET O/P FLAG	= 1 until 1st PET O/P after system start	
C6	G.O.T. FLAG	= 2,1,0 - controls writing date/time,$\overline{sig}$, gains on tape	GAINS-ON-TAPE CHECK-PET
C7	F.G.FLAG	= 1 if file gap required	UPDATE,CHECK-PET
C8	CYCL	controls subroutine choice inOPER2	OPER2
3FE	L	INTERRUPT VECTOR - points to interrupt routine	INITIALIZE
3FF	H		
0	#BLOCKS/RECORD	INITIALIZED MANUALLY AT SYSTEM START. This day # is updated everytime a file gap is put on tape.	INITIALIZE
1	D_{100} DAY#		
2	D_{10}		
3	D_1		
A	L	pointer to one byte below chosen display bytes - INITIALIZED MANUALLY AT START	DISPLAY
B	H		
C	L	USED IN DISPLAY TO POINT TO SCREEN LOCATION	
D	H		

8. Apple program, version 17d (4 antennas)

```
08FC-   20 20 09    JSR   $0920              START
08FF-   4C 75 12    JMP   $1275
0902-   A5 C2       LDA   $C2
0904-   D0 F6       BNE   $08FC
0906-   AD 7A 1F    LDA   $1F7A              start record if M1 = 0 or 5
0909-   F0 04       BEQ   $090F
090B-   C9 05       CMP   #$05
090D-   D0 ED       BNE   $08FC
090F-   E6 BB       INC   $BB
0911-   A9 3C       LDA   #$3C               TURN ON Tx
0913-   8D 01 C5    STA   $C501
0916-   58          CLI
0917-   A5 BB       LDA   $BB                tight loop when record in progress
0919-   D0 FC       BNE   $0917
091B-   78          SEI
091C-   4C FC 08    JMP   $08FC
091F-   00          BRK
0920-   A9 00       LDA   #$00               UPDATE
0922-   20 D0 09    JSR   $09D0
0925-   8D 04 05    STA   $0504   H10
0928-   29 0F       AND   #$0F
092A-   8D 7B 1F    STA   $1F7B
092D-   A9 10       LDA   #$10
092F-   20 D0 09    JSR   $09D0
0932-   8D 05 05    STA   $0505   H1
0935-   29 0F       AND   #$0F
0937-   8D 7C 1F    STA   $1F7C
093A-   A9 BA       LDA   #$BA
093C-   8D 06 05    STA   $0506              read external clock, put time
093F-   A9 20       LDA   #$20               on screen (HR,MIN,SEC),
0941-   20 D0 09    JSR   $09D0              STORE TIME IN "CLOCK"
0944-   8D 07 05    STA   $0507   M10
0947-   29 0F       AND   #$0F
0949-   8D 79 1F    STA   $1F79
094C-   A9 60       LDA   #$60
094E-   20 D0 09    JSR   $09D0
0951-   8D 08 05    STA   $0508   M1
0954-   29 0F       AND   #$0F
0956-   8D 7A 1F    STA   $1F7A
0959-   A9 BA       LDA   #$BA
095B-   8D 09 05    STA   $0509
095E-   A9 50       LDA   #$50
0960-   20 D0 09    JSR   $09D0
0963-   8D 0A 05    STA   $050A   S10
0966-   29 0F       AND   #$0F
0968-   8D 77 1F    STA   $1F77
096B-   A9 40       LDA   #$40
096D-   20 D0 09    JSR   $09D0
0970-   8D 0B 05    STA   $050B   S1
0973-   29 0F       AND   #$0F
0975-   8D 78 1F    STA   $1F78
0978-   AD 7D 1F    LDA   $1F7D
097B-   09 B0       ORA   #$B0    D100       Put day # on screen
097D-   8D 00 05    STA   $0500
0980-   AD 7E 1F    LDA   $1F7E
0983-   09 B0       ORA   #$B0    D10
0985-   8D 01 05    STA   $0501
0988-   AD 7F 1F    LDA   $1F7F
098B-   09 B0       ORA   #$B0    D1
098D-   8D 02 05    STA   $0502
0990-   A9 BA       LDA   #$BA
0992-   8D 03 05    STA   $0503
0995-   EA          NOP
0996-   EA          NOP
0997-   EA          NOP
0998-   A5 BC       LDA   $BC                check for Day # update,
099A-   D0 09       BNE   $09A5              set file gap flag when
099C-   AD 7B 1F    LDA   $1F7B              Day # changes
099F-   C9 02       CMP   #$02
09A1-   D0 22       BNE   $09C5
09A3-   E6 BC       INC   $BC
09A5-   AD 7B 1F    LDA   $1F7B
09A8-   D0 1B       BNE   $09C5
09AA-   E6 C7       INC   $C7
09AC-   A2 03       LDX   #$03
09AE-   BD 7C 1F    LDA   $1F7C,X
```

```
09B1-   18          CLC
09B2-   69 01       ADC   #$01
09B4-   9D 7C 1F    STA   $1F7C,X
09B7-   C9 0A       CMP   #$0A
09B9-   D0 08       BNE   $09C3
09BB-   A9 00       LDA   #$00
09BD-   9D 7C 1F    STA   $1F7C,X
09C0-   CA          DEX
09C1-   D0 EB       BNE   $09AE
09C3-   C6 BC       DEC   $BC
09C5-   A5 BB       LDA   $BB
09C7-   F0 01       BEQ   $09CA        If Record not in progress
09C9-   60          RTS                then "check-pet"
09CA-   20 9E 10    JSR   $109E
09CD-   60          RTS
09CE-   00          BRK
09CF-   00          BRK
09D0-   8D 02 C3    STA   $C302
09D3-   AD 02 C3    LDA   $C302               CNV
09D6-   29 0F       AND   #$0F
09D8-   8D 02 C3    STA   $C302
09DB-   09 B0       ORA   #$B0
09DD-   60          RTS
09DE-   00          BRK
09DF-   00          BRK
09E0-   00          BRK
09E1-   00          BRK
09E2-   00          BRK
09E3-   00          BRK
09E4-   00          BRK
09E5-   00          BRK
09E6-   00          BRK
09E7-   00          BRK
09E8-   00          BRK
09E9-   00          BRK
09EA-   78          SEI                        INITIALIZE
09EB-   D8          CLD
09EC-   A9 07       LDA   #$07
09EE-   A2 20       LDX   #$20
09F0-   A0 04       LDY   #$04
09F2-   9D 7F 1F    STA   $1F7F,X      Set initial gains (arbitrary)
09F5-   CA          DEX                    ht#0-3 have gain 7
09F6-   88          DEY                    ht#4-7 have gain 6
09F7-   D0 F9       BNE   $09F2              etc.
09F9-   38          SEC
09FA-   E9 01       SBC   #$01
09FC-   10 F2       BPL   $09F0
09FE-   A5 00       LDA   $00
0A00-   85 BD       STA   $BD
0A02-   85 BE       STA   $BE
0A04-   A9 39       LDA   #$39         delay to first gate
0A06-   85 B8       STA   $B8
0A08-   A9 08       LDA   #$08         gate separation
0A0A-   85 B9       STA   $B9
0A0C-   A5 01       LDA   $01          load day #
0A0E-   8D 7D 1F    STA   $1F7D
0A11-   A5 02       LDA   $02
0A13-   8D 7E 1F    STA   $1F7E
0A16-   A5 03       LDA   $03
0A18-   8D 7F 1F    STA   $1F7F
0A1B-   A9 14       LDA   #$14         clock in delay
0A1D-   85 BA       STA   $BA
0A1F-   A9 00       LDA   #$00
0A21-   8D 01 C5    STA   $C501        P.I.A. initialization
0A24-   8D 03 C5    STA   $C503
0A27-   8D 05 C5    STA   $C505
0A2A-   8D 07 C5    STA   $C507
0A2D-   8D 01 C7    STA   $C701
0A30-   8D 03 C7    STA   $C703
0A33-   8D 03 C3    STA   $C303
0A36-   A9 FF       LDA   #$FF
0A38-   8D 04 C5    STA   $C504
0A3B-   8D 06 C5    STA   $C506
0A3E-   8D 02 C5    STA   $C502
0A41-   8D 02 C7    STA   $C702
```

```
0A44-   A9 00      LDA   #$00
0A46-   8D 00 C5   STA   $C500
0A49-   8D 00 C7   STA   $C700
0A4C-   A9 F0      LDA   #$F0
0A4E-   8D 02 C3   STA   $C302
0A51-   A9 2D      LDA   #$2D
0A53-   8D 01 C5   STA   $C501
0A56-   8D 05 C5   STA   $C505
0A59-   8D 07 C5   STA   $C507
0A5C-   8D 01 C7   STA   $C701
0A5F-   8D 03 C7   STA   $C703
0A62-   8D 03 C3   STA   $C303
0A65-   8D 03 C5   STA   $C503
0A68-   A0 02      LDY   #$02
0A6A-   A2 20      LDX   #$20
0A6C-   BD 7F 1F   LDA   $1F7F,X
0A6F-   09 C0      ORA   #$C0        Run gain setting into PIA (twice!)
0A71-   8D 02 C7   STA   $C702
0A74-   CA         DEX
0A75-   D0 F5      BNE   $0A6C
0A77-   88         DEY
0A78-   D0 F0      BNE   $0A6A
0A7A-   A0 10      LDY   #$10
0A7C-   A9 00      LDA   #$00        zero date/time store 1
0A7E-   99 EF 5F   STA   $5FEF,Y
0A81-   88         DEY
0A82-   D0 FA      BNE   $0A7E
0A84-   85 BB      STA   $BB
0A86-   85 C2      STA   $C2
0A88-   85 BC      STA   $BC
0A8A-   A9 04      LDA   #$04   <--  4 relays (Antennas)
0A8C-   85 C0      STA   $C0
0A8E-   85 C1      STA   $C1
0A90-   A9 01      LDA   #$01
0A92-   8D 02 C5   STA   $C502
0A95-   A9 C0      LDA   #$C0
0A97-   8D FE 03   STA   $03FE       set interrupt vector
0A9A-   A9 0A      LDA   #$0A
0A9C-   8D FF 03   STA   $03FF
0A9F-   A9 00      LDA   #$00
0AA1-   85 BF      STA   $BF         pulse count
0AA3-   A9 03      LDA   #$03        initial record counter
0AA5-   85 C4      STA   $C4
0AA7-   A9 01      LDA   #$01
0AA9-   85 C5      STA   $C5
0AAB-   A9 00      LDA   #$00
0AAD-   85 C6      STA   $C6
0AAF-   85 C3      STA   $C3
0AB1-   85 C7      STA   $C7
0AB3-   A9 34      LDA   #$34        turn off Tx (in case it was on)
0AB5-   8D 01 C5   STA   $C501
0AB8-   A9 01      LDA   #$01
0ABA-   85 C8      STA   $C8
0ABC-   4C FC 08   JMP   $08FC
0ABF-   00         BRK
0AC0-   48         PHA                          INTERRUPT
0AC1-   8A         TXA
0AC2-   48         PHA
0AC3-   A4 B8      LDY   $B8         delay
0AC5-   88         DEY
0AC6-   D0 FD      BNE   $0AC5
0AC8-   A5 B9      LDA   $B9         turn on clocking at chosen gate sep.
0ACA-   8D 02 C7   STA   $C702
0ACD-   A4 BA      LDY   $BA
0ACF-   A2 0B      LDX   #$0B
0AD1-   CA         DEX               clock in delay
0AD2-   D0 FD      BNE   $0AD1
0AD4-   EA         NOP
0AD5-   88         DEY
0AD6-   D0 F7      BNE   $0ACF
0AD8-   A2 03      LDX   #$03
0ADA-   CA         DEX
0ADB-   D0 FD      BNE   $0ADA
0ADD-   A9 80      LDA   #$80        STOP clocking
0ADF-   8D 02 C7   STA   $C702
```

```
0AE2-   A0 02      LDY   #$02
0AE4-   A2 20      LDX   #$20
0AE6-   BD 7F 1F   LDA   $1F7F,X
0AE9-   09 C0      ORA   #$C0        load gain setting for next pulse
0AEB-   8D 02 C7   STA   $C702
0AEE-   CA         DEX
0AEF-   D0 F5      BNE   $0AE6
0AF1-   88         DEY
0AF2-   D0 F0      BNE   $0AE4
0AF4-   A5 BB      LDA   $BB
0AF6-   D0 03      BNE   $0AFB       if a record is not in progress, skip out
0AF8-   18         CLC
0AF9-   90 5C      BCC   $0B57
0AFB-   D8         CLD
0AFC-   A9 7F      LDA   #$7F
0AFE-   85 A6      STA   $A6
0B00-   A5 BF      LDA   $BF
0B02-   4A         LSR
0B03-   4A         LSR
0B04-   85 A5      STA   $A5         calculate data address
0B06-   A9 00      LDA   #$00
0B08-   A6 C0      LDX   $C0
0B0A-   18         CLC
0B0B-   69 40      ADC   #$40
0B0D-   CA         DEX
0B0E-   D0 FA      BNE   $0B0A
0B10-   18         CLC
0B11-   E5 A5      SBC   $A5
0B13-   85 A5      STA   $A5
0B15-   A0 20      LDY   #$20
0B17-   AD 00 C7   LDA   $C700       read data from PIA and
0B1A-   D0 02      BNE   $0B1E       STORE IN BLOCK STORAGE (N.B. A/D conv
0B1C-   A9 FF      LDA   #$FF        makes anything >255 equal zero  - thus
0B1E-   81 A5      STA   ($A5,X)     this adjustment)
0B20-   C6 A6      DEC   $A6
0B22-   88         DEY
0B23-   D0 F2      BNE   $0B17
0B25-   C6 C0      DEC   $C0
0B27-   D0 04      BNE   $0B2D
0B29-   A5 C1      LDA   $C1
0B2B-   85 C0      STA   $C0
0B2D-   A9 05      LDA   #$05        select next relay
0B2F-   38         SEC
0B30-   E5 C0      SBC   $C0
0B32-   AA         TAX
0B33-   A9 00      LDA   #$00
0B35-   38         SEC
0B36-   2A         ROL
0B37-   CA         DEX
0B38-   D0 FC      BNE   $0B36
0B3A-   8D 02 C5   STA   $C502  <--  Relay #1 2 3 4
0B3D-   A5 BE      LDA   $BE         Loaded#1,2,4,8
0B3F-   C5 BD      CMP   $BD
0B41-   D0 06      BNE   $0B49
0B43-   20 70 0B   JSR   $0B70       choose subroutine depending on Block #
0B46-   18         CLC
0B47-   90 03      BCC   $0B4C
0B49-   20 2C 0C   JSR   $0C2C
0B4C-   E6 BF      INC   $BF
0B4E-   D0 07      BNE   $0B57
0B50-   C6 BD      DEC   $BD
0B52-   D0 03      BNE   $0B57
0B54-   20 75 0C   JSR   $0C75
0B57-   20 20 09   JSR   $0920       UPDATE
0B5A-   20 5D 12   JSR   $125D       DISPLAY
0B5D-   AD 02 C7   LDA   $C702  <--  RESET INTERRUPT enable on PIA
0B60-   68         PLA
0B61-   AA         TAX
0B62-   68         PLA
0B63-   40         RTI
0B64-   00         BRK
0B65-   00         BRK
0B66-   00         BRK
0B67-   00         BRK
0B68-   00         BRK
0B69-   00         BRK
0B6A-   00         BRK
0B6B-   00         BRK
0B6C-   00         BRK
0B6D-   00         BRK
0B6E-   00         BRK
0B6F-   00         BRK
```

```
0B70-   D8          CLD                        OPER 1 (done during 1st block
0B71-   A5 BF       LDA   $BF      check pulse        in record)
0B73-   D0 24       BNE   $0B99    count
0B75-   A0 10       LDY   #$10
0B77-   B9 EF 5F    LDA   $5FEF,Y  shift store 1
0B7A-   99 DF 43    STA   $43DF,Y  date/time
0B7D-   A9 00       LDA   #$00     to store 2 & zero store 1
0B7F-   99 EF 5F    STA   $5FEF,Y
0B82-   88          DEY
0B83-   D0 F2       BNE   $0B77
0B85-   A9 5E       LDA   #$5E
0B87-   85 A2       STA   $A2
0B89-   A9 F0       LDA   #$F0     shift first 14 pages store 1 to store 2
0B8B-   85 A1       STA   $A1      (auto & cross) and zero store 1
0B8D-   A9 42       LDA   #$42
0B8F-   85 A4       STA   $A4
0B91-   A9 E0       LDA   #$E0
0B93-   85 A3       STA   $A3
0B95-   20 14 0C    JSR   $0C14
0B98-   60          RTS
0B99-   C9 01       CMP   #$01
0B9B-   D0 16       BNE   $0BB3    shift and zero last 14 pages
0B9D-   20 14 0C    JSR   $0C14
0BA0-   A5 C4       LDA   $C4
0BA2-   F0 06       BEQ   $0BAA
0BA4-   C6 C4       DEC   $C4      check record count
0BA6-   C9 03       CMP   #$03
0BA8-   F0 08       BEQ   $0BB2
0BAA-   A9 02       LDA   #$02 <-- gains on tape flag
0BAC-   85 C6       STA   $C6
0BAE-   A9 20       LDA   #$20
0BB0-   85 C2       STA   $C2
0BB2-   60          RTS
0BB3-   C9 02       CMP   #$02
0BB5-   D0 42       BNE   $0BF9
0BB7-   A0 40       LDY   #$40
0BB9-   A9 00       LDA   #$00
0BBB-   99 9F 1F    STA   $1F9F,Y  zero Σsig store
0BBE-   88          DEY
0BBF-   D0 FA       BNE   $0BBB
0BC1-   A9 E0       LDA   #$E0
0BC3-   85 A1       STA   $A1
0BC5-   A9 26       LDA   #$26
0BC7-   85 A2       STA   $A2      zero binary store 1 & 2
0BC9-   A2 08       LDX   #$08
0BCB-   A9 00       LDA   #$00
0BCD-   A0 00       LDY   #$00
0BCF-   91 A1       STA   ($A1),Y
0BD1-   88          DEY
0BD2-   D0 FB       BNE   $0BCF
0BD4-   C6 A2       DEC   $A2
0BD6-   CA          DEX
0BD7-   D0 F4       BNE   $0BCD
0BD9-   A2 07       LDX   #$07
0BDB-   BD 78 1F    LDA   $1F78,X
0BDE-   9D F8 5F    STA   $5FF8,X  load present time ( & other parameters)
0BE1-   CA          DEX            into date/time store 1
0BE2-   D0 F7       BNE   $0BDB
0BE4-   A5 BE       LDA   $BE
0BE6-   8D F8 5F    STA   $5FF8
0BE9-   A5 B8       LDA   $B8
0BEB-   8D F7 5F    STA   $5FF7
0BEE-   A5 B9       LDA   $B9
0BF0-   8D F6 5F    STA   $5FF6
0BF3-   A5 BA       LDA   $BA
0BF5-   8D F5 5F    STA   $5FF5
0BF8-   60          RTS
0BF9-   C9 FC       CMP   #$FC
0BFB-   B0 04       BCS   $0C01
0BFD-   20 9E 10    JSR   $109E    CHECK-PET
0C00-   60          RTS
0C01-   38          SEC
0C02-   E9 FC       SBC   #$FC
0C04-   85 A0       STA   $A0 <--- ANT #
0C06-   A9 00       LDA   #$00 <-- HT #
0C08-   85 A1       STA   $A1           (Convert decimal to binary as each
0C0A-   20 C8 0C    JSR   $0CC8 <-- ADR-BIN antenna is finished in the block
0C0D-   20 78 0D    JSR   $0D78 <-- DEC-BIN
0C10-   60          RTS
```

```
0C11-   00          BRK
0C12-   00          BRK
0C13-   00          BRK
0C14-   A2 0E       LDX   #$0E                        MOVE 14
0C16-   A0 00       LDY   #$00
0C18-   B1 A1       LDA   ($A1),Y           Move 14 pages store 1 auto & cross
0C1A-   91 A3       STA   ($A3),Y           correls to store 2 and zero store 1
0C1C-   A9 00       LDA   #$00
0C1E-   91 A1       STA   ($A1),Y
0C20-   88          DEY
0C21-   D0 F5       BNE   $0C18
0C23-   C6 A2       DEC   $A2
0C25-   C6 A4       DEC   $A4
0C27-   CA          DEX
0C28-   D0 EC       BNE   $0C16
0C2A-   60          RTS
0C2B-   00          BRK
0C2C-   A5 BF       LDA   $BF                         OPER 2
0C2E-   C9 60       CMP   #$60
0C30-   B0 28       BCS   $0C5A
0C32-   C6 C8       DEC   $C8
0C34-   D0 04       BNE   $0C3A
0C36-   A9 03       LDA   #$03
0C38-   85 C8       STA   $C8
0C3A-   A5 C8       LDA   $C8
0C3C-   C9 02       CMP   #$02
0C3E-   F0 0F       BEQ   $0C4F
0C40-   90 1C       BCC   $0C5E             after first 3x32 pulses,
0C42-   C6 9E       DEC   $9E               cycle around XCOR,ACOR,check-pet,
0C44-   A5 9E       LDA   $9E               using index C8 to pick operation
0C46-   85 A1       STA   $A1
0C48-   20 C8 0C    JSR   $0CC8     ADR-BIN
0C4B-   20 27 0E    JSR   $0E27     XCOR
0C4E-   60          RTS
0C4F-   A5 9E       LDA   $9E
0C51-   85 A1       STA   $A1
0C53-   20 C8 0C    JSR   $0CC8     ADR-BIN
0C56-   20 38 0F    JSR   $0F38     ACOR
0C59-   60          RTS
0C5A-   C9 FC       CMP   #$FC
0C5C-   B0 04       BCS   $0C62
0C5E-   20 9E 10    JSR   $109E     CHECK-PET
0C61-   60          RTS
0C62-   38          SEC
0C63-   E9 FC       SBC   #$FC              convert decimal to binary
0C65-   85 A0       STA   $A0
0C67-   A9 00       LDA   #$00
0C69-   85 A1       STA   $A1
0C6B-   20 C8 0C    JSR   $0CC8
0C6E-   20 78 0D    JSR   $0D78
0C71-   60          RTS
0C72-   00          BRK
0C73-   00          BRK
0C74-   00          BRK
0C75-   A9 02       LDA   #$02                        OPER 3
0C77-   85 9E       STA   $9E
0C79-   A9 1F       LDA   #$1F      Run over XCOR & ACOR twice to finish up
0C7B-   48          PHA             HT# counter
0C7C-   85 A1       STA   $A1
0C7E-   20 C8 0C    JSR   $0CC8
0C81-   EA          NOP
0C82-   EA          NOP
0C83-   EA          NOP
0C84-   20 27 0E    JSR   $0E27
0C87-   68          PLA
0C88-   85 A1       STA   $A1
0C8A-   48          PHA
0C8B-   20 C8 0C    JSR   $0CC8
0C8E-   EA          NOP
0C8F-   EA          NOP
0C90-   EA          NOP
0C91-   20 38 0F    JSR   $0F38
0C94-   68          PLA
0C95-   38          SEC
0C96-   E9 01       SBC   #$01
0C98-   10 E1       BPL   $0C7B
```

```
0C9A-   C6 9E       DEC   $9E
0C9C-   D0 DB       BNE   $0C79
0C9E-   20 2C 10    JSR   $102C        RESET GAINS
0CA1-   A0 02       LDY   #$02
0CA3-   A2 20       LDX   #$20
0CA5-   BD 7F 1F    LDA   $1F7F,X
0CA8-   09 C0       ORA   #$C0
0CAA-   8D 02 C7    STA   $C702        LOAD P.I.A. with NEW GAINS
0CAD-   CA          DEX
0CAE-   D0 F5       BNE   $0CA5
0CB0-   88          DEY
0CB1-   D0 F0       BNE   $0CA3
0CB3-   A9 00       LDA   #$00
0CB5-   85 BB       STA   $BB
0CB7-   A5 BE       LDA   $BE
0CB9-   85 BD       STA   $BD
0CBB-   A9 34       LDA   #$34         TURN OFF Tx
0CBD-   8D 01 C5    STA   $C501
0CC0-   60          RTS
0CC1-   00          BRK
0CC2-   00          BRK
0CC3-   00          BRK
0CC4-   00          BRK
0CC5-   00          BRK
0CC6-   00          BRK
0CC7-   00          BRK
0CC8-   A5 A1       LDA   $A1 <---- HT#          ADR-BIN
0CCA-   4A          LSR
0CCB-   4A          LSR
0CCC-   85 AD       STA   $AD                    Calculate address for
0CCE-   0A          ASL                          binary storage, given
0CCF-   0A          ASL                          HT#
0CD0-   85 A2       STA   $A2
0CD2-   A9 27       LDA   #$27
0CD4-   38          SEC
0CD5-   E5 AD       SBC   $AD
0CD7-   85 AD       STA   $AD
0CD9-   A9 A7       LDA   #$A7
0CDB-   85 AC       STA   $AC
0CDD-   A5 A1       LDA   $A1
0CDF-   38          SEC
0CE0-   E5 A2       SBC   $A2
0CE2-   F0 0F       BEQ   $0CF3
0CE4-   AA          TAX
0CE5-   A5 AC       LDA   $AC
0CE7-   38          SEC
0CE8-   E9 40       SBC   #$40
0CEA-   85 AC       STA   $AC
0CEC-   B0 02       BCS   $0CF0
0CEE-   C6 AD       DEC   $AD
0CF0-   CA          DEX
0CF1-   D0 F4       BNE   $0CE7
0CF3-   60          RTS
0CF4-   00          BRK
0CF5-   00          BRK
0CF6-   00          BRK
0CF7-   00          BRK
0CF8-   00          BRK
0CF9-   00          BRK
0CFA-   00          BRK
0CFB-   00          BRK
0CFC-   00          BRK
0CFD-   00          BRK
0CFE-   A5 A1       LDA   $A1 <---- HT#          ADR-XCOR
0D00-   4A          LSR
0D01-   4A          LSR                          Calculate address for
0D02-   4A          LSR                          accumulating Xcorrels
0D03-   0A          ASL                          given HT#
0D04-   0A          ASL
0D05-   85 AF       STA   $AF
0D07-   0A          ASL
0D08-   85 A2       STA   $A2
0D0A-   A5 AF       LDA   $AF
0D0C-   4A          LSR
0D0D-   18          CLC
0D0E-   65 AF       ADC   $AF
```

```
0D10-   85 AF      STA   $AF
0D12-   A9 5F      LDA   #$5F
0D14-   38         SEC
0D15-   E5 AF      SBC   $AF
0D17-   85 AF      STA   $AF
0D19-   A9 AF      LDA   #$AF
0D1B-   85 AE      STA   $AE
0D1D-   A5 A1      LDA   $A1
0D1F-   38         SEC
0D20-   E5 A2      SBC   $A2
0D22-   F0 0F      BEQ   $0D33
0D24-   AA         TAX
0D25-   A5 AE      LDA   $AE
0D27-   38         SEC
0D28-   E9 C0      SBC   #$C0
0D2A-   85 AE      STA   $AE
0D2C-   B0 02      BCS   $0D30
0D2E-   C6 AF      DEC   $AF
0D30-   CA         DEX
0D31-   D0 F4      BNE   $0D27
0D33-   60         RTS
```

```
0D34-   00         BRK
0D35-   00         BRK
0D36-   00         BRK
0D37-   00         BRK
0D38-   00         BRK
0D39-   00         BRK
0D3A-   00         BRK
0D3B-   00         BRK
0D3C-   00         BRK
0D3D-   00         BRK
0D3E-   00         BRK
0D3F-   00         BRK
```

ADR-ACOR

Calculate address for accumulating auto correls given ht#

```
0D40-   A5 A1      LDA   $A1
0D42-   4A         LSR
0D43-   4A         LSR
0D44-   4A         LSR
0D45-   85 AB      STA   $AB
0D47-   0A         ASL
0D48-   0A         ASL
0D49-   0A         ASL
0D4A-   85 A2      STA   $A2
0D4C-   A9 47      LDA   #$47
0D4E-   38         SEC
0D4F-   E5 AB      SBC   $AB
0D51-   85 AB      STA   $AB
0D53-   A9 CF      LDA   #$CF
0D55-   85 AA      STA   $AA
0D57-   A5 A1      LDA   $A1
0D59-   38         SEC
0D5A-   E5 A2      SBC   $A2
0D5C-   F0 0F      BEQ   $0D6D
0D5E-   AA         TAX
0D5F-   A5 AA      LDA   $AA
0D61-   38         SEC
0D62-   E9 20      SBC   #$20
0D64-   85 AA      STA   $AA
0D66-   B0 02      BCS   $0D6A
0D68-   C6 AB      DEC   $AB
0D6A-   CA         DEX
0D6B-   D0 F4      BNE   $0D61
0D6D-   60         RTS
```

```
0D6E-   00         BRK
0D6F-   00         BRK
0D70-   00         BRK
0D71-   00         BRK
0D72-   00         BRK
0D73-   00         BRK
0D74-   00         BRK
0D75-   00         BRK
0D76-   00         BRK
0D77-   00         BRK
```

DEC-BIN (convert 1 Ant to binary)

re-adjust address to binary

```
0D78-   A5 AC      LDA   $AC
0D7A-   38         SEC
0D7B-   E9 08      SBC   #$08
0D7D-   85 AC      STA   $AC
0D7F-   B0 02      BCS   $0D83
```

```
0D81-   C6 AD      DEC  $AD
0D83-   A9 7F      LDA  #$7F
0D85-   85 A2      STA  $A2
0D87-   A9 BF      LDA  #$BF
0D89-   85 A1      STA  $A1
0D8B-   A5 A0      LDA  $A0
0D8D-   F0 0F      BEQ  $0D9E
0D8F-   AA         TAX
0D90-   A5 A1      LDA  $A1
0D92-   38         SEC
0D93-   E9 40      SBC  #$40        — calculate data address
0D95-   85 A1      STA  $A1
0D97-   B0 02      BCS  $0D9B
0D99-   C6 A2      DEC  $A2
0D9B-   CA         DEX
0D9C-   D0 F4.     BNE  $0D92
0D9E-   A9 03      LDA  #$03
0DA0-   38         SEC
0DA1-   E5 A0      SBC  $A0
0DA3-   F0 0F      BEQ  $0DB4
0DA5-   A8         TAY              — calculate binary address(ht#0) for
0DA6-   A5 AC      LDA  $AC            given ant#
0DA8-   18         CLC
0DA9-   69 10      ADC  #$10
0DAB-   85 AC      STA  $AC
0DAD-   90 02      BCC  $0DB1
0DAF-   E6 AD      INC  $AD
0DB1-   88         DEY
0DB2-   D0 F4      BNE  $0DA8
0DB4-   A9 20      LDA  #$20        — ht#
0DB6-   85 A3      STA  $A3
0DB8-   A0 40      LDY  #$40
0DBA-   A2 00      LDX  #$00        — add 64 amplitudes
0DBC-   A9 00      LDA  #$00
0DBE-   18         CLC
0DBF-   71 A1      ADC  ($A1),Y
0DC1-   90 01      BCC  $0DC4
0DC3-   E8         INX
0DC4-   88         DEY
0DC5-   D0 F7      BNE  $0DBE
0DC7-   4A         LSR              — ÷by 64 = sig
0DC8-   4A         LSR
0DC9-   4A         LSR
0DCA-   4A         LSR
0DCB-   4A         LSR
0DCC-   4A         LSR
0DCD-   85 A7      STA  $A7
0DCF-   8A         TXA
0DD0-   0A         ASL
0DD1-   0A         ASL
0DD2-   38         SEC  <—————————— intended
0DD3-   65 A7      ADC  $A7
0DD5-   85 A7      STA  $A7
0DD7-   A5 A3      LDA  $A3
0DD9-   EA         NOP              — calculate index for Σsig store
0DDA-   EA         NOP
0DDB-   EA         NOP
0DDC-   0A         ASL
0DDD-   AA         TAX
0DDE-   A5 A7      LDA  $A7
0DE0-   18         CLC
0DE1-   7D 9F 1F   ADC  $1F9F,X
0DE4-   9D 9F 1F   STA  $1F9F,X     — add to Σsig store
0DE7-   90 04      BCC  $0DED
0DE9-   CA         DEX
0DEA-   FE 9F 1F   INC  $1F9F,X
0DED-   A0 40      LDY  #$40        — convert 64 bytes to
0DEF-   A9 08      LDA  #$08           binary
0DF1-   85 A6      STA  $A6
0DF3-   A9 00      LDA  #$00        convert 8 data bytes
0DF5-   85 A4      STA  $A4         to one "binary byte"
0DF7-   A2 08      LDX  #$08
0DF9-   B1 A1      LDA  ($A1),Y
0DFB-   C5 A7      CMP  $A7
0DFD-   66 A4      ROR  $A4
0DFF-   88         DEY
```

```
0E00-   CA          DEX
0E01-   D0 F6       BNE   $0DF9
0E03-   84 A5       STY   $A5
0E05-   A4 A6       LDY   $A6
0E07-   A5 A4       LDA   $A4
0E09-   91 AC       STA   ($AC),Y
0E0B-   A4 A5       LDY   $A5
0E0D-   C6 A6       DEC   $A6
0E0F-   D0 E2       BNE   $0DF3
0E11-   A5 AC       LDA   $AC        adjust addresses for next ht#
0E13-   38          SEC
0E14-   E9 40       SBC   #$40
0E16-   85 AC       STA   $AC
0E18-   B0 02       BCS   $0E1C
0E1A-   C6 AD       DEC   $AD
0E1C-   C6 A2       DEC   $A2
0E1E-   C6 A3       DEC   $A3
0E20-   D0 96       BNE   $0DB8
0E22-   A9 20       LDA   #$20       used in OPER 2 for selecting ht#
0E24-   85 9E       STA   $9E        for XCOR & ACOR
0E26-   60          RTS
-----------------------------------------------------------------
0E27-   20 FE 0C    JSR   $0CFE                 XCOR
0E2A-   A9 A1       LDA   #$A1
0E2C-   85 B0       STA   $B0
0E2E-   A9 00       LDA   #$00
0E30-   85 B1       STA   $B1
0E32-   A9 08       LDA   #$08
0E34-   85 9F       STA   $9F
0E36-   A5 AD       LDA   $AD
0E38-   85 B3       STA   $B3
0E3A-   A5 AC       LDA   $AC
0E3C-   38          SEC
0E3D-   E9 02       SBC   #$02
0E3F-   EA          NOP
0E40-   EA          NOP
0E41-   85 B2       STA   $B2
0E43-   B0 02       BCS   $0E47
0E45-   C6 B3       DEC   $B3
0E47-   A0 0A       LDY   #$0A
0E49-   B1 B2       LDA   ($B2),Y    load Ant# 4 binary (10bytes) into zero
0E4B-   91 B0       STA   ($B0),Y    page for "comparing" and rolling
0E4D-   88          DEY
0E4E-   D0 F9       BNE   $0E49
0E50-   A9 00       LDA   #$00       zero 2 bytes below this
0E52-   85 A0       STA   $A0          (not required)
0E54-   85 A1       STA   $A1
0E56-   A9 40       LDA   #$40       lag x 2 #
0E58-   85 B7       STA   $B7
0E5A-   A9 03       LDA   #$03       Xcorrel #
0E5C-   85 B6       STA   $B6
0E5E-   A5 AE       LDA   $AE
0E60-   85 B2       STA   $B2
0E62-   A5 AF       LDA   $AF
0E64-   85 B3       STA   $B3
0E66-   A5 AD       LDA   $AD
0E68-   85 B5       STA   $B5
0E6A-   A5 AC       LDA   $AC
0E6C-   18          CLC
0E6D-   69 10       ADC   #$10
0E6F-   85 B4       STA   $B4
0E71-   90 02       BCC   $0E75
0E73-   E6 B5       INC   $B5        = eight for -ve lags
0E75-   A4 9F       LDY   $9F        = ten for +ve lags
0E77-   A2 00       LDX   #$00
0E79-   B1 B0       LDA   ($B0),Y
0E7B-   F0 22       BEQ   $0E9F
0E7D-   31 B4       AND   ($B4),Y
0E7F-   4A          LSR
0E80-   90 01       BCC   $0E83      add the #1's in the result of the "AND",
0E82-   E8          INX              LINEAR program to save time.
0E83-   4A          LSR
0E84-   90 01       BCC   $0E87
0E86-   E8          INX
0E87-   4A          LSR
0E88-   90 01       BCC   $0E8B
```

```
0E8A-   E8          INX
0E8B-   4A          LSR
0E8C-   90 01       BCC   $0E8F
0E8E-   E8          INX
0E8F-   4A          LSR
0E90-   90 01       BCC   $0E93
0E92-   E8          INX
0E93-   4A          LSR
0E94-   90 01       BCC   $0E97
0E96-   E8          INX
0E97-   4A          LSR
0E98-   90 01       BCC   $0E9B
0E9A-   E8          INX
0E9B-   4A          LSR
0E9C-   90 01       BCC   $0E9F
0E9E-   E8          INX
0E9F-   88          DEY
0EA0-   D0 D7       BNE   $0E79
0EA2-   A4 B7       LDY   $B7
0EA4-   8A          TXA
0EA5-   18          CLC
0EA6-   71 B2       ADC   ($B2),Y ┐
0EA8-   91 B2       STA   ($B2),Y ├─add #1's to Xcorrel store 1 at proper lag
0EAA-   90 08       BCC   $0EB4   │
0EAC-   88          DEY           │
0EAD-   18          CLC           │
0EAE-   B1 B2       LDA   ($B2),Y │
0EB0-   69 01       ADC   ##01    │
0EB2-   91 B2       STA   ($B2),Y ┘
0EB4-   A5 B4       LDA   $B4
0EB6-   18          CLC
0EB7-   69 10       ADC   #$10
0EB9-   85 B4       STA   $B4
0EBB-   90 02       BCC   $0EBF
0EBD-   E6 B5       INC   $B5
0EBF-   A5 B2       LDA   $B2
0EC1-   38          SEC
0EC2-   E9 40       SBC   #$40
0EC4-   85 B2       STA   $B2
0EC6-   B0 02       BCS   $0ECA
0EC8-   C6 B3       DEC   $B3
0ECA-   C6 B6       DEC   $B6
0ECC-   D0 A7       BNE   $0E75
0ECE-   C6 B7       DEC   $B7
0ED0-   C6 B7       DEC   $B7
0ED2-   F0 3B       BEQ   $0F0F
0ED4-   18          CLC
0ED5-   26 AB       ROL   $AB     ┐
0ED7-   26 AA       ROL   $AA     │
0ED9-   26 A9       ROL   $A9     │
0EDB-   26 A8       ROL   $A8     ├─shift ANT#4 zero page sequence to
0EDD-   26 A7       ROL   $A7     │ the left by 1 bit
0EDF-   26 A6       ROL   $A6     │
0EE1-   26 A5       ROL   $A5     │
0EE3-   26 A4       ROL   $A4     │
0EE5-   26 A3       ROL   $A3     │
0EE7-   26 A2       ROL   $A2     │
0EE9-   26 A1       ROL   $A1     │
0EEB-   26 A0       ROL   $A0     ┘
0EED-   A5 B7       LDA   $B7
0EEF-   C9 20       CMP   #$20
0EF1-   D0 19       BNE   $0F0C
0EF3-   A9 0A       LDA   #$0A    ┐
0EF5-   85 9F       STA   $9F     ├─ adjust addresses etc for +ve lags
0EF7-   A9 9F       LDA   #$9F    │
0EF9-   85 B0       STA   $B0     │
0EFB-   A9 00       LDA   #$00    │
0EFD-   85 A0       STA   $A0     │
0EFF-   85 A1       STA   $A1     │
0F01-   A5 AC       LDA   $AC     ┘
0F03-   38          SEC
0F04-   E9 02       SBC   #$02
0F06-   85 AC       STA   $AC
0F08-   B0 02       BCS   $0F0C
```

```
0F0A-   C6 AD       DEC   $AD
0F0C-   4C 5A 0E    JMP   $0E5A
0F0F-   A5 AC       LDA   $AC
0F11-   18          CLC
0F12-   69 02       ADC   #$02        Calculate addressing for
0F14-   85 AC       STA   $AC         following operation
0F16-   90 02       BCC   $0F1A
0F18-   E6 AD       INC   $AD
0F1A-   A5 AD       LDA   $AD
0F1C-   85 B5       STA   $B5
0F1E-   A5 AC       LDA   $AC
0F20-   38          SEC
0F21-   E9 08       SBC   #$08
0F23-   85 B4       STA   $B4
0F25-   B0 02       BCS   $0F29
0F27-   C6 B5       DEC   $B5
0F29-   A0 08       LDY   #$08
0F2B-   B1 B4       LDA   ($B4),Y
0F2D-   91 AC       STA   ($AC),Y     Transfer ANT#4 store 1 binary
0F2F-   A9 00       LDA   #$00        to store 2 & zero store 1
0F31-   91 B4       STA   ($B4),Y
0F33-   88          DEY
0F34-   D0 F5       BNE   $0F2B
0F36-   60          RTS
0F37-   00          BRK
0F38-   20 40 0D    JSR   $0D40                  ACOR  (calculate & Auto correls
0F3B-   A5 AD       LDA   $AD                          for ANT# 1,2,3)
0F3D-   85 AF       STA   $AF
0F3F-   A5 AC       LDA   $AC
0F41-   18          CLC
0F42-   69 0E       ADC   #$0E        Calculate addresses to binary store
0F44-   85 AE       STA   $AE
0F46-   90 02       BCC   $0F4A
0F48-   E6 AF       INC   $AF
0F4A-   A5 AF       LDA   $AF
0F4C-   85 B3       STA   $B3
0F4E-   A5 AE       LDA   $AE
0F50-   18          CLC
0F51-   69 02       ADC   #$02
0F53-   85 B2       STA   $B2
0F55-   90 02       BCC   $0F59
0F57-   E6 B3       INC   $B3
0F59-   A9 03       LDA   #$03        auto correl # (ANT#)
0F5B-   85 B4       STA   $B4
0F5D-   A9 A1       LDA   #$A1
0F5F-   85 B0       STA   $B0
0F61-   A9 00       LDA   #$00
0F63-   85 B1       STA   $B1
0F65-   A0 08       LDY   #$08
0F67-   B1 B2       LDA   ($B2),Y     transfer sequence to zero page
0F69-   91 B0       STA   ($B0),Y
0F6B-   88          DEY
0F6C-   D0 F9       BNE   $0F67
0F6E-   A9 9F       LDA   #$9F        modify zero page address and zero
0F70-   85 B0       STA   $B0         two bytes below
0F72-   A9 00       LDA   #$00
0F74-   85 A0       STA   $A0
0F76-   85 A1       STA   $A1
0F78-   A9 20       LDA   #$20
0F7A-   85 B5       STA   $B5
0F7C-   A9 00       LDA   #$00
0F7E-   85 B6       STA   $B6
0F80-   A0 0A       LDY   #$0A        "AND" ten bytes
0F82-   B1 B0       LDA   ($B0),Y
0F84-   F0 0C       BEQ   $0F92
0F86-   31 AE       AND   ($AE),Y
0F88-   A2 08       LDX   #$08        add #1's in result of "AND" operation
0F8A-   4A          LSR
0F8B-   90 02       BCC   $0F8F
0F8D-   E6 B6       INC   $B6
0F8F-   CA          DEX
0F90-   D0 F8       BNE   $0F8A
0F92-   88          DEY
0F93-   D0 ED       BNE   $0F82
0F95-   A4 B5       LDY   $B5
0F97-   B1 AA       LDA   ($AA),Y
0F99-   18          CLC
```

```
0F9A-   65 B6       ADC   $B6
0F9C-   91 AA       STA   ($AA),Y
0F9E-   90 08       BCC   $0FA8
0FA0-   88          DEY
0FA1-   B1 AA       LDA   ($AA),Y
0FA3-   18          CLC
0FA4-   69 01       ADC   #$01
0FA6-   91 AA       STA   ($AA),Y
0FA8-   18          CLC
0FA9-   26 A9       ROL   $A9
0FAB-   26 A8       ROL   $A8
0FAD-   26 A7       ROL   $A7
0FAF-   26 A6       ROL   $A6
0FB1-   26 A5       ROL   $A5
0FB3-   26 A4       ROL   $A4
0FB5-   EA          NOP
0FB6-   EA          NOP                shift left by 1 bit
0FB7-   26 A3       ROL   $A3
0FB9-   26 A2       ROL   $A2
0FBB-   26 A1       ROL   $A1
0FBD-   26 A0       ROL   $A0
0FBF-   C6 B5       DEC   $B5
0FC1-   C6 B5       DEC   $B5
0FC3-   D0 B7       BNE   $0F7C
0FC5-   A5 AE       LDA   $AE
0FC7-   18          CLC
0FC8-   69 10       ADC   #$10
0FCA-   85 AE       STA   $AE
0FCC-   90 02       BCC   $0FD0        reset addresses for next ant#
0FCE-   E6 AF       INC   $AF
0FD0-   A5 B2       LDA   $B2
0FD2-   18          CLC
0FD3-   69 10       ADC   #$10
0FD5-   85 B2       STA   $B2
0FD7-   90 02       BCC   $0FDB
0FD9-   E6 B3       INC   $B3
0FDB-   C6 B4       DEC   $B4
0FDD-   F0 03       BEQ   $0FE2
0FDF-   4C 5D 0F    JMP   $0F5D
0FE2-   A5 B2       LDA   $B2
0FE4-   38          SEC
0FE5-   E9 10       SBC   #$10
0FE7-   85 B2       STA   $B2
0FE9-   B0 02       BCS   $0FED
0FEB-   C6 B3       DEC   $B3
0FED-   A5 B3       LDA   $B3
0FEF-   85 AF       STA   $AF          modify addresses for following
0FF1-   A5 B2       LDA   $B2          operation
0FF3-   38          SEC
0FF4-   E9 08       SBC   #$08
0FF6-   85 AE       STA   $AE
0FF8-   B0 02       BCS   $0FFC
0FFA-   C6 AF       DEC   $AF
0FFC-   A9 03       LDA   #$03
0FFE-   85 B4       STA   $B4
1000-   A0 08       LDY   #$08
1002-   B1 AE       LDA   ($AE),Y
1004-   91 B2       STA   ($B2),Y
1006-   A9 00       LDA   #$00
1008-   91 AE       STA   ($AE),Y
100A-   88          DEY
100B-   D0 F5       BNE   $1002        move store 2 binary to store 1 binary
100D-   38          SEC                and zero store 2 binary for ANT#1,2,3
100E-   A5 B2       LDA   $B2
1010-   E9 10       SBC   #$10
1012-   85 B2       STA   $B2
1014-   B0 02       BCS   $1018
1016-   C6 B3       DEC   $B3
1018-   38          SEC
1019-   A5 AE       LDA   $AE
101B-   E9 10       SBC   #$10
101D-   85 AE       STA   $AE
101F-   B0 02       BCS   $1023
1021-   C6 AF       DEC   $AF
1023-   C6 B4       DEC   $B4
1025-   D0 D9       BNE   $1000
1027-   60          RTS
1028-   00          BRK
1029-   00          BRK
102A-   00          BRK
102B-   00          BRK
```

```
102C-   A9 20      LDA   #$20  }-- ht#             RESET-GAINS
102E-   85 A3      STA   $A3   }
1030-   A9 00      LDA   #$00
1032-   85 A1      STA   $A1
1034-   85 A0      STA   $A0
1036-   A5 BE      LDA   $BE
1038-   0A         ASL
1039-   0A         ASL
103A-   85 A2      STA   $A2   <------ # BLOCKS X4
103C-   A2 08      LDX   #$08
103E-   18         CLC
103F-   66 A2      ROR   $A2
1041-   66 A1      ROR   $A1                          X #ANTS
1043-   A5 A3      LDA   $A3
1045-   0A         ASL         -Divide Σsig by # blocks (truncates)
1046-   A8         TAY
1047-   38         SEC
1048-   B9 9F 1F   LDA   $1F9F,Y
104B-   E5 A1      SBC   $A1
104D-   85 A4      STA   $A4
104F-   88         DEY
1050-   B9 9F 1F   LDA   $1F9F,Y
1053-   E5 A2      SBC   $A2
1055-   90 09      BCC   $1060
1057-   99 9F 1F   STA   $1F9F,Y
105A-   C8         INY
105B-   A5 A4      LDA   $A4
105D-   99 9F 1F   STA   $1F9F,Y
1060-   26 A0      ROL   $A0
1062-   CA         DEX
1063-   D0 D9      BNE   $103E
1065-   A6 A3      LDX   $A3     }
1067-   BD 7F 1F   LDA   $1F7F,X }-save old gains for O/P to tape
106A-   9D 36 1F   STA   $1F36,X }
106D-   A5 A0      LDA   $A0
106F-   C9 50      CMP   #$50
1071-   B0 0D      BCS   $1080
1073-   BD 7F 1F   LDA   $1F7F,X   -Reset-gains
1076-   C9 07      CMP   #$07         if 80≤sig≤160 leave gain as is,
1078-   F0 12      BEQ   $108C        otherwise try to move it up or
107A-   FE 7F 1F   INC   $1F7F,X      down by one step
107D-   18         CLC
107E-   90 0C      BCC   $108C
1080-   C9 A0      CMP   #$A0
1082-   90 08      BCC   $108C
1084-   BD 7F 1F   LDA   $1F7F,X
1087-   F0 03      BEQ   $108C
1089-   DE 7F 1F   DEC   $1F7F,X
108C-   A6 A3      LDX   $A3
108E-   A5 A0      LDA   $A0     }
1090-   9D 56 1F   STA   $1F56,X }-save sig for O/P to tape
1093-   C6 A3      DEC   $A3
1095-   D0 99      BNE   $1030
1097-   60         RTS
1098-   00         BRK
1099-   A5 C4      LDA   $C4
109B-   C9 02      CMP   #$02                CHECK-PET
109D-   D0 01      BNE   $10A0
109F-   60         RTS                    (see flow chart)
10A0-   A5 C2      LDA   $C2              controls data O/P to and I/P
10A2-   D0 01      BNE   $10A5            from PET, and tape recorder
10A4-   60         RTS                    (data, block gaps, file gaps)
10A5-   A5 C3      LDA   $C3
10A7-   F0 14      BEQ   $10BD
10A9-   A9 11      LDA   #$11
10AB-   8D 04 C5   STA   $C504
10AE-   A9 00      LDA   #$00
10B0-   8D 04 C5   STA   $C504
10B3-   AD 00 C5   LDA   $C500
10B6-   C9 F0      CMP   #$F0
10B8-   F0 F9      BEQ   $10B3
10BA-   C6 C3      DEC   $C3
10BC-   60         RTS
10BD-   A5 C2      LDA   $C2
10BF-   C9 1F      CMP   #$1F
```

```
10C1-   D0 08       BNE   $10CB
10C3-   A5 C6       LDA   $C6
10C5-   F0 04       BEQ   $10CB
10C7-   20 3A 11    JSR   $113A
10CA-   60          RTS
10CB-   AD 01 C3    LDA   $C301
10CE-   29 80       AND   #$80
10D0-   D0 01       BNE   $10D3
10D2-   60          RTS
10D3-   A9 00       LDA   #$00
10D5-   8D 01 C3    STA   $C301
10D8-   8D 00 C3    STA   $C300
10DB-   A9 3C       LDA   #$3C
10DD-   8D 01 C3    STA   $C301
10E0-   20 A0 11    JSR   $11A0
10E3-   A5 C2       LDA   $C2
10E5-   C9 20       CMP   #$20
10E7-   D0 27       BNE   $1110
10E9-   A5 C5       LDA   $C5
10EB-   F0 05       BEQ   $10F2
10ED-   C6 C5       DEC   $C5
10EF-   18          CLC
10F0-   90 21       BCC   $1113
10F2-   A5 C4       LDA   $C4
10F4-   D0 1A       BNE   $1110
10F6-   A5 C7       LDA   $C7
10F8-   F0 14       BEQ   $110E
10FA-   AD EB 43    LDA   $43EB
10FD-   D0 0F       BNE   $110E
10FF-   C6 C7       DEC   $C7
1101-   20 85 11    JSR   $1185
1104-   20 1A 11    JSR   $111A
1107-   EA          NOP
1108-   EA          NOP
1109-   EA          NOP
110A-   EA          NOP
110B-   18          CLC
110C-   90 05       BCC   $1113
110E-   E6 C3       INC   $C3
1110-   20 85 11    JSR   $1185
1113-   20 CA 11    JSR   $11CA
1116-   C6 C2       DEC   $C2
1118-   60          RTS
1119-   00          BRK
111A-   A9 12       LDA   #$12                  FILE-GAP
111C-   8D 04 C5    STA   $C504
111F-   A9 00       LDA   #$00
1121-   8D 04 C5    STA   $C504
1124-   AD 00 C5    LDA   $C500
1127-   C9 F0       CMP   #$F0
1129-   F0 F9       BEQ   $1124
112B-   A2 03       LDX   #$03
112D-   BD 7C 1F    LDA   $1F7C,X    "update" initial day #
1130-   95 00       STA   $00,X
1132-   CA          DEX
1133-   D0 F8       BNE   $112D
1135-   60          RTS
1136-   00          BRK
1137-   00          BRK
1138-   00          BRK
1139-   00          BRK
113A-   A5 C6       LDA   $C6                   GAINS-TO-TAPE
113C-   C9 01       CMP   #$01
113E-   F0 27       BEQ   $1167
1140-   A2 10       LDX   #$10
1142-   BD DF 43    LDA   $43DF,X    put date/time store 2 on tape
1145-   8D 06 C5    STA   $C506
1148-   AD 00 C5    LDA   $C500                      Batch 1 O/P
114B-   C9 E0       CMP   #$E0
114D-   F0 F9       BEQ   $1148
114F-   CA          DEX
1150-   D0 F0       BNE   $1142
1152-   A2 20       LDX   #$20
1154-   BD 56 1F    LDA   $1F56,X    put sig on tape
1157-   8D 06 C5    STA   $C506
115A-   AD 00 C5    LDA   $C500
115D-   C9 E0       CMP   #$E0
115F-   F0 F9       BEQ   $115A
```

```
1161-   CA          DEX
1162-   D0 F0       BNE   $1154
1164-   C6 C6       DEC   $C6
1166-   60          RTS
1167-   A2 20       LDX   #$20
1169-   BD 36 1F    LDA   $1F36,X                 "Batch 2" O/P
116C-   8D 06 C5    STA   $C506
116F-   AD 00 C5    LDA   $C500      put gain settings on tape
1172-   C9 E0       CMP   #$E0
1174-   F0 F9       BEQ   $116F
1176-   CA          DEX
1177-   D0 F0       BNE   $1169
1179-   C6 C6       DEC   $C6
117B-   60          RTS
117C-   00          BRK
117D-   00          BRK
117E-   00          BRK
117F-   00          BRK
1180-   00          BRK
1181-   00          BRK
1182-   00          BRK
1183-   00          BRK
1184-   00          BRK
1185-   A2 10       LDX   #$10                    TAPE 16
1187-   BD 26 1F    LDA   $1F26,X
118A-   8D 06 C5    STA   $C506                    Put 16 bytes (pet O/P)
118D-   AD 00 C5    LDA   $C500                    on tape
1190-   C9 E0       CMP   #$E0
1192-   F0 F9       BEQ   $118D
1194-   CA          DEX
1195-   D0 F0       BNE   $1187
1197-   60          RTS
1198-   00          BRK
1199-   00          BRK
119A-   00          BRK
119B-   00          BRK
119C-   00          BRK
119D-   00          BRK
119E-   00          BRK
119F-   00          BRK
11A0-   A2 10       LDX   #$10
11A2-   AD 01 C3    LDA   $C301                   READ 16
11A5-   29 80       AND   #$80
11A7-   F0 F9       BEQ   $11A2                    Accept 16 bytes from Pet
11A9-   AD 00 C3    LDA   $C300
11AC-   9D 26 1F    STA   $1F26,X
11AF-   A9 3C       LDA   #$3C
11B1-   8D 01 C3    STA   $C301
11B4-   A9 34       LDA   #$34
11B6-   8D 01 C3    STA   $C301
11B9-   CA          DEX
11BA-   D0 E6       BNE   $11A2
11BC-   AD 00 C3    LDA   $C300
11BF-   60          RTS
11C0-   00          BRK
11C1-   00          BRK
11C2-   00          BRK
11C3-   00          BRK
11C4-   00          BRK
11C5-   00          BRK
11C6-   00          BRK
11C7-   00          BRK
11C8-   00          BRK
11C9-   00          BRK
11CA-   A9 20       LDA   #$20
11CC-   38          SEC                           WRITE 240
11CD-   E5 C2       SBC   $C2                      Transmit  240 bytes to PET
11CF-   85 A1       STA   $A1  ←———HT#
11D1-   8D E4 43    STA   $43E4 ←——put ht# in date/time store 2 as data is
11D4-   A9 00       LDA   #$00      O/P
11D6-   8D 01 C3    STA   $C301
11D9-   A9 FF       LDA   #$FF
11DB-   8D 00 C3    STA   $C300
11DE-   A9 3C       LDA   #$3C
```

```
11E0-   8D 01 C3   STA   $C301
11E3-   A2 10      LDX   #$10         ┐
11E5-   BD DF 43   LDA   $43DF,X      │
11E8-   8D 00 C3   STA   $C300        ├─date/time store 2 O/P
11EB-   20 3D 12   JSR   $123D        │
11EE-   CA         DEX                │
11EF-   D0 F4      BNE   $11E5        ┘
11F1-   20 FE 0C   JSR   $0CFE        ┐
11F4-   A5 AE      LDA   $AE          │
11F6-   38         SEC                │
11F7-   E9 90      SBC   #$90         │
11F9-   85 AE      STA   $AE          │
11FB-   B0 02      BCS   $11FF        │
11FD-   C6 AF      DEC   $AF          │                            ⊖
11FF-   A5 AF      LDA   $AF          │
1201-   38         SEC                ├─Xcorrel O/P
1202-   E9 1C      SBC   #$1C         │ (calculate address from ADR XCOR and
1204-   85 AF      STA   $AF          │ adjust for store 2 (Auto & Cross))
1206-   A0 C0      LDY   #$C0         │
1208-   B1 AE      LDA   ($AE),Y      │
120A-   8D 00 C3   STA   $C300        │
120D-   20 3D 12   JSR   $123D        │
1210-   88         DEY                │
1211-   D0 F5      BNE   $1208        │
1213-   20 40 0D   JSR   $0D40        ┘
1216-   A5 AA      LDA   $AA          ┐
1218-   38         SEC                │
1219-   E9 10      SBC   #$10         │
121B-   85 AA      STA   $AA          │
121D-   B0 02      BCS   $1221        │
121F-   C6 AB      DEC   $AB          │
1221-   A5 AB      LDA   $AB          ├─Auto correl O/P
1223-   38         SEC                │
1224-   E9 1C      SBC   #$1C         │
1226-   85 AB      STA   $AB          │
1228-   A0 20      LDY   #$20         │
122A-   B1 AA      LDA   ($AA),Y      │
122C-   8D 00 C3   STA   $C300        │
122F-   20 3D 12   JSR   $123D        │
1232-   88         DEY                │
1233-   D0 F5      BNE   $122A        │
1235-   60         RTS                ┘
-------------------------------------------------------------
1236-   00         BRK
1237-   00         BRK
1238-   00         BRK
1239-   00         BRK
123A-   00         BRK
123B-   00         BRK
123C-   00         BRK
-------------------------------------------------------------
123D-   A9 3C      LDA   #$3C                       PIA-INIT
123F-   8D 01 C3   STA   $C301
1242-   A9 34      LDA   #$34
1244-   8D 01 C3   STA   $C301
1247-   AD 01 C3   LDA   $C301
124A-   29 80      AND   #$80
124C-   F0 F9      BEQ   $1247
124E-   AD 00 C3   LDA   $C300
1251-   60         RTS
-------------------------------------------------------------
1252-   00         BRK
1253-   00         BRK
1254-   00         BRK
1255-   00         BRK
1256-   00         BRK
1257-   00         BRK
1258-   00         BRK
1259-   00         BRK
125A-   00         BRK
125B-   00         BRK
125C-   00         BRK
-------------------------------------------------------------
125D-   A9 0C      LDA   #$0C                       DISPLAY
125F-   85 0C      STA   $0C                        PUT 17 "bytes" ON SCREEN
1261-   A9 05      LDA   #$05
1263-   85 0D      STA   $0D                        address ØA,ØB must be set
1265-   A0 11      LDY   #$11                       manually when program is
1267-   B1 0A      LDA   ($0A),Y                    loaded
1269-   29 0F      AND   #$0F
```

```
126B-   09 B0        ORA   ##$B0
126D-   91 0C        STA   ($0C),Y
126F-   88           DEY
1270-   D0 F5        BNE   $1267
1272-   60           RTS
1273-   00           BRK
1274-   00           BRK
1275-   20 BA 12     JSR   $12BA                         WAIT (NOT A SUBROUTINE)
1278-   D0 03        BNE   $127D
127A-   4C 02 09     JMP   $0902                         For manual operations
127D-   C9 D7        CMP   ##$D7 ——— "W"                 while system is running
127F-   F0 03        BEQ   $1284
1281-   4C 02 09     JMP   $0902
1284-   8D 0C 05     STA   $050C ———store "W" on screen
1287-   20 BA 12     JSR   $12BA
128A-   C9 C5        CMP   ##$C5 ——— "E"
128C-   D0 06        BNE   $1294
128E-   20 D7 12     JSR   $12D7                              W= WAIT
1291-   4C 87 12     JMP   $1287                              E=END OF TAPE
~~1294-   20 BA 12     JSR   $12BA~~                              S= START
1297-   C9 D3        CMP   ##$D3 ——— "S",                        (ie return from
1299-   D0 08        BNE   $12A3                                  "W")
129B-   A9 A0        LDA   ##$A0 ]                            M= initiate manual
129D-   8D 0C 05     STA   $050C ]— "b"                          record
12A0-   4C 02 09     JMP   $0902
12A3-   C9 CD        CMP   ##$CD ——— "M"
12A5-   D0 E0        BNE   $1287
12A7-   A9 A0        LDA   ##$A0 ]
12A9-   8D 0C 05     STA   $050C ]— "b"
12AC-   4C 0F 09     JMP   $090F
12AF-   00           BRK
12B0-   00           BRK
12B1-   00           BRK
12B2-   00           BRK
12B3-   00           BRK
12B4-   00           BRK
12B5-   00           BRK
12B6-   00           BRK
12B7-   00           BRK
12B8-   00           BRK
12B9-   00           BRK
12BA-   20 20 09     JSR   $0920 <——"update"          KEY
12BD-   A9 00        LDA   ##$00
12BF-   2C 00 C0     BIT   $C000                  if no key pressed then
12C2-   10 08        BPL   $12CC                  accum = 0
12C4-   AD 00 C0     LDA   $C000
12C7-   2C 10 C0     BIT   $C010
12CA-   09 80        ORA   ##$80
12CC-   60           RTS
12CD-   00           BRK
12CE-   00           BRK
12CF-   00           BRK
12D0-   00           BRK
12D1-   00           BRK
12D2-   00           BRK
12D3-   00           BRK
12D4-   00           BRK
12D5-   00           BRK
12D6-   00           BRK
12D7-   A2 10        LDX   ##$10                    END OF TAPE
12D9-   A9 00        LDA   ##$00
12DB-   8D 06 C5     STA   $C506                  put 16 bytes (0's) and a
12DE-   AD 00 C5     LDA   $C500                  file gap on tape
12E1-   C9 E0        CMP   ##$E0
12E3-   F0 F9        BEQ   $12DE
12E5-   CA           DEX
12E6-   D0 F3        BNE   $12DB
12E8-   20 1A 11     JSR   $111A
12EB-   A9 02        LDA   ##$02
12ED-   85 C4        STA   $C4
12EF-   A9 01        LDA   ##$01
12F1-   85 C5        STA   $C5
12F3-   60           RTS
12F4-   00           BRK
12F5-   00           BRK
```

APPENDIX B

Theory of binary correlation

1. Calculation of correlation coefficient from number of 1-matches
2. Correction to binary correlation (Gaussian amplitude distribution)
3. Effect of overloading on correction function (Gaussian)
4. Test of correction function on Rayleigh distributed data
5. Further comments on the binary method

1. Calculation of correlation from # 1-matches

The correlation coefficient is defined by:

$$\rho = \frac{\overline{xy} - \bar{x}\,\bar{y}}{\sqrt{(\overline{x^2} - \bar{x}^{\,2})(\,\overline{y^2} - \bar{y}^{\,2})}}$$

where x and y are elements of the same (auto correlation) or different (cross correlation) fading sequences, and there may be some lag between x and y measurements.

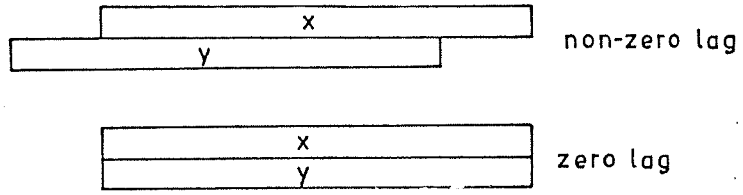


Fig. 1

If the elements x and y are binary (i.e. 0 or 1), then $\sum xy$ is given directly by the results of the 'AND' operation; and so $\overline{xy} = (\#\ 1\text{-matches})/(N-\ell)$, where N is the total number of samples in a sequence and ℓ is the lag. Also $\bar{x} = \sum x/N$, where $\sum x$ is the number of ones in the sequence ; and $\overline{x^2} = \sum x^2/N$, where $\sum x^2$ is again the number of ones (since x can only be 0 or 1) . The number of 1's could be counted in each sequence separately, however, since the fading should be statistically the same at each antenna, the sum of the #1-matches from the zero lag auto correlation calculations can be used to estimate the 'average' number of 1's in a sequence. For various reasons this sum has only be calculated over the three outer antennas (#1,2,3). Since the amplitude distribution is usually skewed towards low amplitudes (Rayleigh type), the number of 1's is usually $< \frac{1}{2}N$. Thus, if a_o is the number of 1-matches ($\sum xy$), and a_1 is the number of 1's in any sequence ($\sum x$), the correlation coefficient is:

$$\rho' = \frac{\dfrac{a_o}{N-\ell} - \dfrac{a_1^2}{N^2}}{\dfrac{a_1}{N} - \dfrac{a_1^2}{N^2}}$$

This is the correlation between binary sequences, and must be corrected as explained below, to estimate the correlation which would have been obtained with the original sequences.

2. Correction to binary correlation

As will be shown below, the expectation value of correlation between two amplitude sequences which have been converted to binary with respect to their means is lower than that which would have been found between the original sequences. Thus some sort of correction is required. The derivation of this correction function closely follows that of Weinreb(1963). A basic assumption is that the amplitudes have an approximately Gaussian (i.e. Normal) distribution. This is not generally true in practice - usually the distribution is closer to a Rayleigh - however, it makes the problem solvable. The binary correlation, as employed in the present analysis, counts the number of times that both X and Y amplitudes are greater than their means. The probability of this event will be derived next.

Assume that X and Y are taken from a Normal distribution with zero mean and unit s.d. Then the joint probability density function for X and Y is (e.g. Hald, 1952):

$$f(x,y) = \frac{1}{2\pi\sqrt{1-\rho^2}} \exp\left\{-\frac{1}{2(1-\rho^2)}(x^2 -2\rho xy + y^2)\right\}$$

where ρ is the correlation coefficient between X and Y. Thus the probability of finding X between x and x+dx and Y between y and y+dy simultaneously is f(x,y)dxdy. The probability of finding X and Y both greater than their means is then:

$$P(X>0,Y>0) = \int_{x=0}^{\infty}\int_{y=0}^{\infty} f(x,y)dydx$$

If polar coordinates are used (x=r cosθ , y=r sinθ) the integral is reduced to

$$P(X>0,\ Y>0) = \frac{1}{2\pi\sqrt{1-\rho^2}}\int_{\theta=0}^{\frac{\pi}{2}}\int_{r=0}^{\infty}\exp\left\{\frac{-r^2(1-\rho\sin 2\theta)}{2(1-\rho^2)}\right\} r\,dr\,d\theta$$

A further change of variable $\xi = r^2$, $d\xi = 2r\,dr$, and integration of the inner integral leaves:

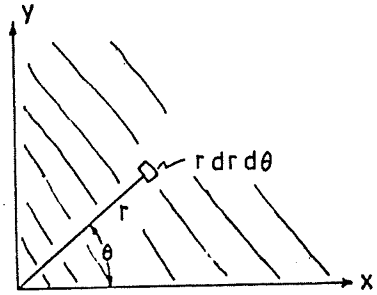


Fig. 1

$$P(X>0,\ Y>0) = \frac{\sqrt{1-\rho^2}}{2\pi}\int_{\theta=0}^{\frac{\pi}{2}} \frac{1}{(1-\rho\sin 2\theta)}\,d\theta$$

Then let $\alpha = 2\theta$:

$$P(X>0,\ Y>0) = \frac{\sqrt{1-\rho^2}}{2}\int_{\alpha=0}^{\frac{\pi}{2}} \frac{1}{1-\rho\sin\alpha}\,d\alpha$$

$$= \frac{1}{\pi}\tan^{-1}\left\{\frac{\tan\frac{\alpha}{2}-\rho}{\sqrt{1-\rho^2}}\right\}\Bigg|_{\alpha=0}^{\frac{\pi}{2}}$$

(This last integral is taken from math tables). The identity

$$\tan^{-1}(a) - \tan^{-1}(b) = \tan^{-1}\left(\frac{a-b}{1-ab}\right)$$

is used to obtain:

$$P(X>0,\ Y>0) = \frac{1}{\pi}\tan^{-1}\sqrt{\frac{1+\rho}{1-\rho}}$$

Now $P(X>0, Y>0)$ must be related to the correlation of the binary sequences X' and Y'. These are defined by:

$$X' = 1 \text{ if } X>0 \qquad Y' = 1 \text{ if } Y>0$$
$$X' = 0 \text{ if } X\leq 0 \qquad Y' = 0 \text{ if } Y\leq 0$$

Figure 2 shows the joint probability function of X', Y'. Here $d = P(X>0,\ Y>0)$. Since a Gaussian distribution has been assumed, the marginal probabilities are $P(X>0)=P(X\leq 0)=P(Y>0)=P(Y\leq 0)=\frac{1}{2}$. (It is easy to show that a=d and b=c).

	X'=		
	0	1	tot.
Y'= 0	a	c	$\frac{1}{2}$
1	b	d	$\frac{1}{2}$
tot.	$\frac{1}{2}$	$\frac{1}{2}$	

Fig. 2

The cross correlation between X' and Y' is then:

$$\rho' = \frac{\overline{X'\,Y'} - \overline{X'}\;\overline{Y'}}{\sqrt{\overline{X'^2} - \overline{X'}^2}\sqrt{\overline{Y'^2} - \overline{Y'}^2}} = \frac{d-(\frac{1}{2})^2}{\sqrt{(\frac{1}{2}) - (\frac{1}{2})^2}\sqrt{(\frac{1}{2}) - (\frac{1}{2})^2}} = 4d-1$$

Thus:

$$\rho' = \frac{4}{\pi} \tan^{-1}\sqrt{\frac{1+\rho}{1-\rho}} - 1$$

This can be re-arranged to give

$$\rho = \sin \frac{\pi}{2} \rho'$$

This then is the correction function which must be applied to the result of correlating the binary representations of sequences in order to estimate the actual correlation between the original sequences.

3. Effect of overloading on the correlation correction

It is of interest to inquire whether receiver overloading will affect the correction function. This may also indicate whether less major receiver non-linearity will affect the results.

The receiver output for an input signal causing overloading will be assumed to be the maximum output value. It can be seen (Fig. 1) that $\bar{A}$, the mean amplitude, is less than the actual mean of the Gaussian. Thus the binary sequence will have more ones than zeroes, and thus a mean $> \frac{1}{2}$. Alternatively non-linearity of the type shown in Fig.2 will skew the distribution towards low amplitudes (assuming no overloading occurs) resulting in a mean $< \frac{1}{2}$. Such skewing occurs naturally in a Rayleigh distribution, and a later section will try to answer the present question in this case.

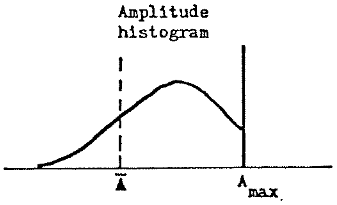


Fig. 1

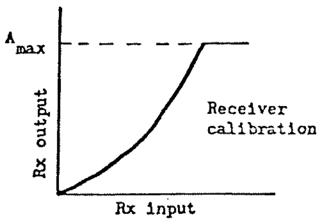


Fig. 2

Here, the procedure is to choose a value, ϵ, and find $P(X>\epsilon, Y>\epsilon)$ for various values of ρ (the actual correlation). ϵ is negative in the case of overloading. Fig. 3 shows contours of $f(x,y)$ for ρ = +ve, 0, and -ve; and the region (shaded) which must be integrated over to find $P(X>\epsilon\,,\ Y>\epsilon\,)$.

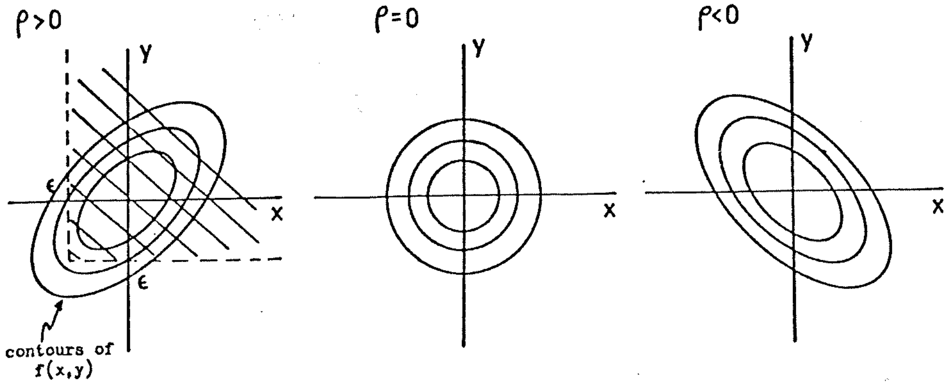


Fig. 3

It is assumed that the Gaussians (for X and Y) have zero mean and unit s.d. as before (there is no loss of generality in this assumption). Thus the mean of the binary sequences, m, is the area of the normal curve (from statistical tables) above ϵ (see Fig. 4), and the binary correlation is:

$$\rho' = \frac{P(X>\epsilon\,, Y>\epsilon\,) - m^2}{m(1-m)}$$

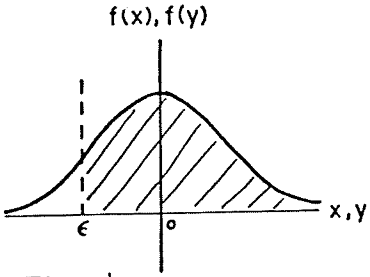


Fig. 4

where: $P = \int_{x=\epsilon}^{\infty}\int_{y=\epsilon}^{\infty} f(x,y)\,dydx$. Unfortunately when $\epsilon \neq 0$ the integral does not reduce to a simple solution, and has been performed numerically here (The upper limits used were x=y=8, so the P values will be slightly low). Values of ϵ between ±1 were used; however, instead of quoting ϵ , m is given since it is directly available in the experimental situation. Fig. 5 shows a plot of ρ vs. $\sin\frac{\pi}{2}\rho'$ for +ve ρ and several values of m. The difference between the ideal and overloaded data

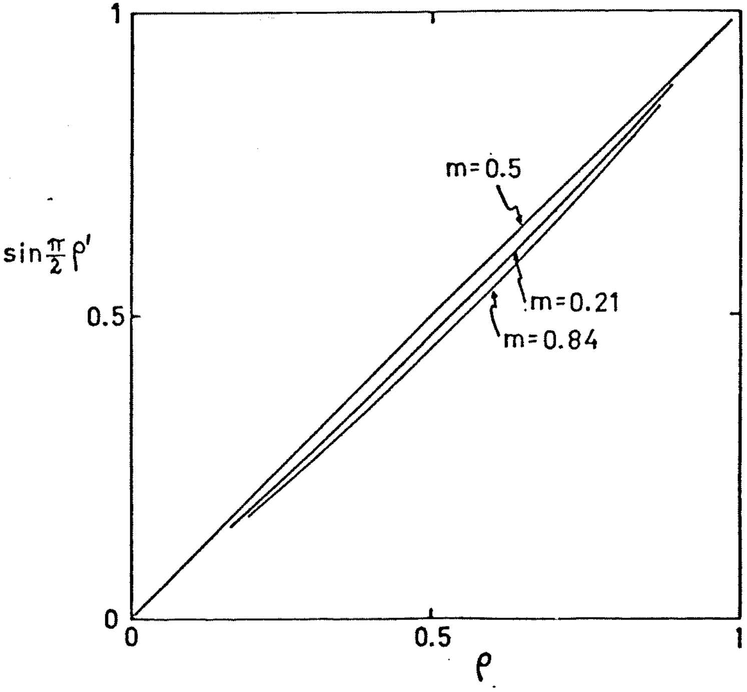


Fig. 5

(m=0.84) is seen to be almost negligible, and not worth correcting for.

In practice, however, overloading may have serious effects on the correlation if, as is done here, the sequences are assumed to be statistically equivalent (i.e. equal m). This problem will be discussed in Section 5.

4. Test with a Rayleigh distribution

In order to see whether the correction function works when the data are Rayleigh distributed, pairs of sequences X and Y were constructed as follows:

$$X_i = \sqrt{(a_i + K\,e_i)^2 + (b_i + K\,f_i)^2}$$
$$Y_i = \sqrt{(c_i + K\,e_i)^2 + (d_i + K\,f_i)^2}$$

where a_i, b_i, etc. are independent random values taken from a Gaussian distribution. This definition results in a non-zero expected correlation between X and Y while leaving them Rayleigh distributed. The expectation value of correlation depends on K, which has been chosen to be 2 for this test. 200 pairs of sequences (100 points each) were set up and correlated before and after conversion to binary (with respect to the mean). Histograms of the ratio of the actual to corrected binary correlations , and the actual correlations are shown in Fig. 1. Although there is some spread in the corrected values, there is no bias away from a ratio of 1. Thus the

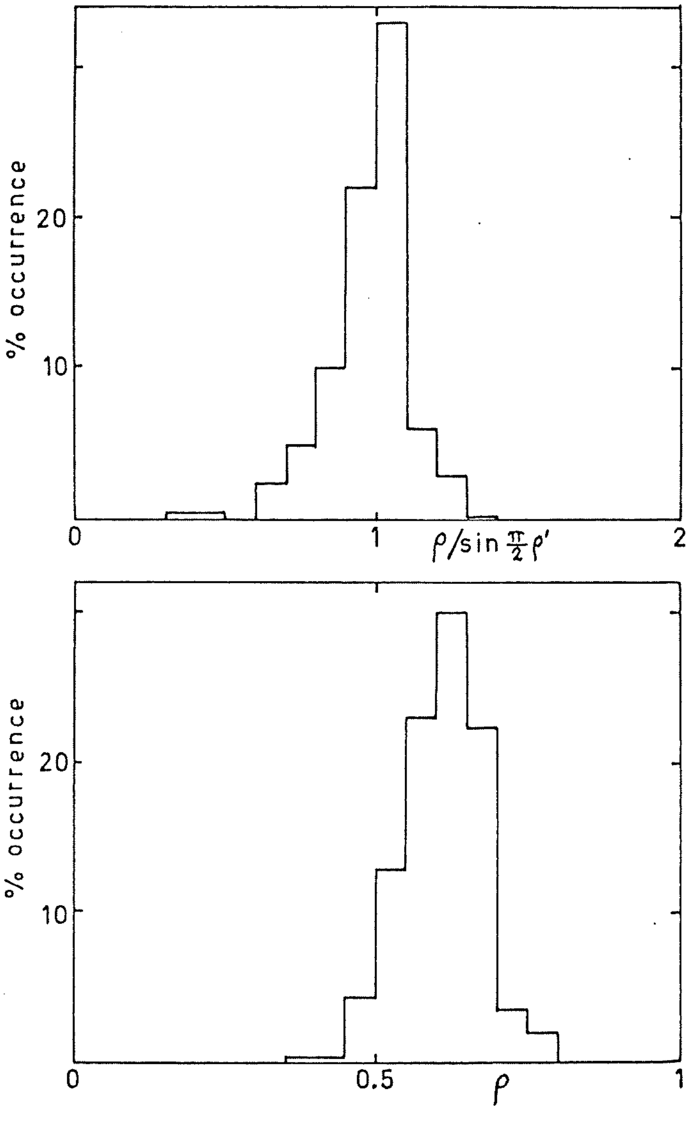


Fig. 1

same correction function should work for Rayleigh distributed data as well.

5. Further comments on binary correlation

a) The binary correlation, ρ' , has been defined as:

$$\rho'_{ij} = \frac{P_{ij} - m^2}{m(1-m)}$$

where m is the 'average' mean of the binary sequences over all antennas (in practice just #1,2,3). This is not a strictly correct definition; it should be

$$\rho'_{ij} = \frac{P_{ij} - m_i m_j}{\sqrt{m_i(1-m_i)m_j(1-m_j)}}$$

where ρ'_{ij} is the correlation between binary sequences from antenna i and j; however, if ρ' is not very sensitive to changes in m, the first definition should be sufficient since all antennas should see similar amplitude distributions (and therefore similar values of m). A plot of ρ' (and $\rho = \sin \frac{\pi}{2}\rho'$) versus P and m is shown in Fig. 1.

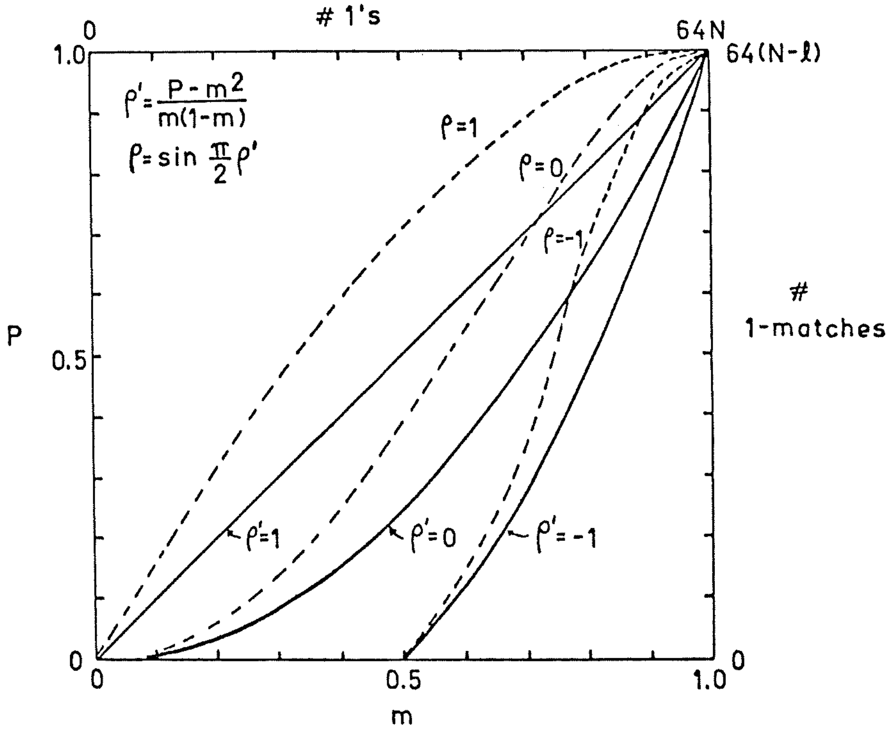


Fig. 1

Note that P cannot be $> m$ since the # 1-matches cannot be greater than the number of 1's in any sequence; also, if $m > 0.5$, P must be $> 2m^2-m$, because if both sequences are more than half 1's, there must be some 1-matches.

For reasonably Gaussian data (or any data with a symmetrical distribution) $m \cong 0.5$, and ρ' is relatively insensitive to small errors in m. However, if there is serious overloading, m will generally increase and Fig. 1 shows that ρ' becomes more sensitive to errors in m (and to errors in P for that matter, although P is a strictly correct value and will only have errors in a statistical sense). If, in addition, the amplitude distributions at different antennas are not strictly the same then the small error caused by the use of m rather than m_i, m_j may move the whole correlation curve up or down (with only small changes in shape) depending on whether m is less or greater than $\sqrt{m_i m_j}$. This shifting is seen quite often in badly overloaded data (Fig. 2) although a direct connection to differences in m_i, m_j etc. has not been proven as yet.

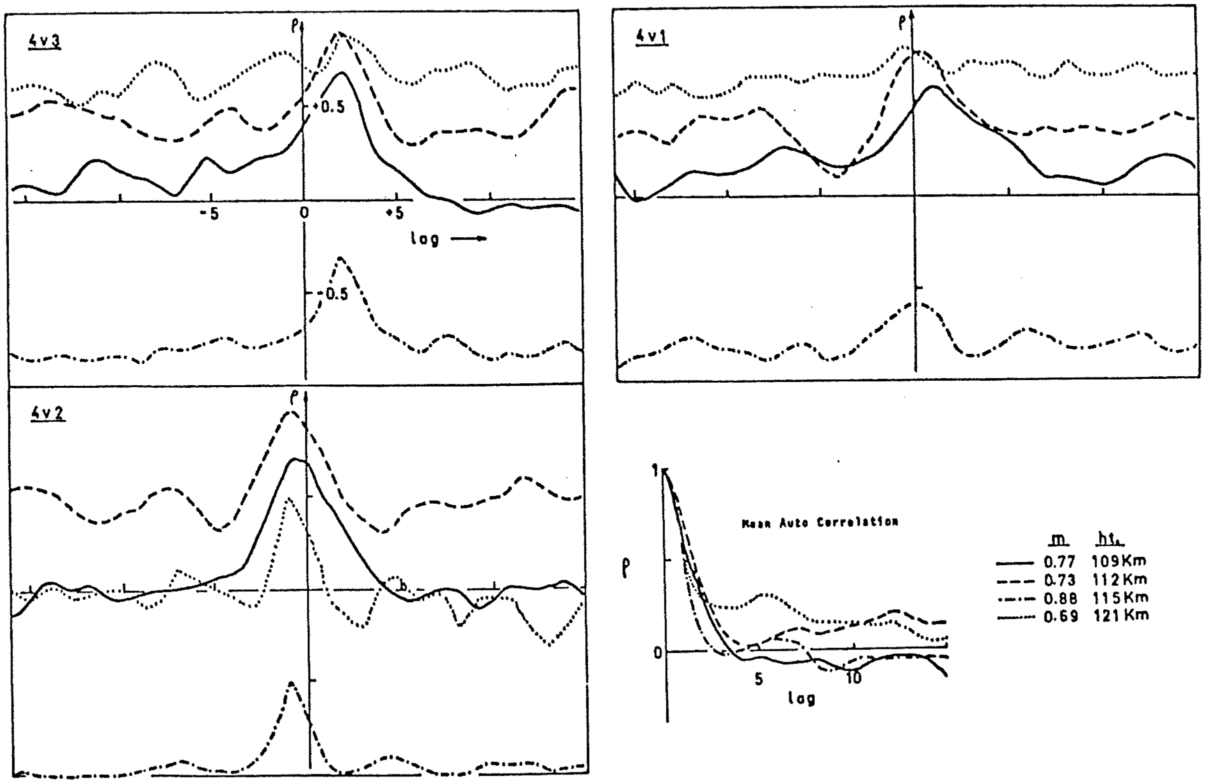


Fig. 2

In a computor simulation of the binary correlation process on real data which used separate sequence means, this problem didn't occur at all. On a separate point, if m_i and m_j are different but m is used to find ρ', then values > 1 or < -1 are possible; however, the correction through the sin function will reduce the limits to ±1.

Other information which can be deduced from Fig. 1 is the expected resolution in ρ' (or ρ); how likely are exactly equal adjacent values of ρ (see the discussion in Appendix C, Section 1, pg.69 on locating peaks in cross correlations). The number of 1-matches is given by $P(N-\ell)$, and the number of 1's by mN (where N is the number of points in a sequence and ℓ is the lag - since $\ell << N$, it may be ignored). For example if N=512 (8 blocks/record) and m=0.5, there are 256 possible values of correlation which gives an average resolution of~0.008 in ρ' . If m=0.8 (i.e. bad overloading), there are 154 values, giving an average resolution of 0.14.

The conclusion is that overloading (or any other receiver non-linearity which causes amplitude distributions to be skewed towards high values) is quite serious, and should be avoided if at all possible. On the other hand skewing the distribution in the other direction (i.e. low m) is quite tolerable, and perhaps desirable, since ρ' becomes less sensitive to m here; and it has already been shown that the $\sin \frac{\pi}{2}\rho'$ correction works on at least one distribution of this type (i.e. Rayleigh).

If, as has been postulated, the sensitivity of ρ' to m is the cause of the trouble shown in Fig. 2, then a solution which tolerates overloading would be to use individual means m_i instead of the average mean m in the calculation of ρ' . From the point of view of the Apple program, the auto correlation routine (ACOR) would have to be modified to operate on all four antennas, and the zero lag values would have to be accumulated separately for each antenna. This appears to be feasible from an execution time standpoint and would not require major modifications to the program.

b) It has been noticed by simulating the binary correlation on raw data from the previous wind system, that the correlation correction seems to work well below ∽80 Km, but results in an average reduction in peak values of ∽1.5-2 below the normal (non-binary calculation) correlation peaks above ∽80 Km. Presumably this has something to do with a change in amplitude distribution - possibly related to the effect of meteor trails at these heights. Investigation proceeds...

APPENDIX C

Pet program details

1. Description of Pet analysis
2. Correction to time delays for antenna cycling
3. Weighted least squares fit to time delays
4. Poor man's full correlation analysis (PMFCA)
5. Machine language subroutine
6. Input/output memory locations
7. Listing of BASIC program

1. Pet analysis

The input to the Pet computer is the number of 1-matches at each of 32 lags (-16 to +15) for the three cross correlations (c_{ij}; i=1,32, j=1,3) and the sum (over three antennas) of the number of 1-matches for 16 lags (0 to 15) for the auto correlation (A_i; i=0,15). The conversion of these values to correct correlation values has been described in Appendix B (pg. 58) and the relevant equations are repeated below.

$$\rho = \sin \frac{\pi}{2} \rho'$$

where: $$\rho' = \frac{\frac{c_{ij}}{N-\ell} - m^2}{m(1-m)} \qquad \text{(cross correlation)}$$

or: $$\rho' = \frac{\frac{a_i}{N-\ell} - m^2}{m(1-m)} \qquad \text{(auto correlation)}$$

where $a_i=A_i/3$, $m=a_o/N$, N= # points in each fading sequence (i.e. N = number of blocks per record times 64), and ℓ is the absolute value of the lag. Here ρ' is the actual correlation value for the binary sequences and ρ is the corrected value. In the Pet analysis usually ρ' is used in place of ρ until an accurate value is required, since time is at a premium.

The program (see listing on pg. 79) falls into nine sections, which are described below.

a) Initialization

D is the antenna spacing (assumed equal for all antenna pairs); P1,P2,P3 are the directions of the antenna pairs 4v3 (in the sense 4 to 3 since t_{43} is being used), 4v2, and 4v1. TS is the sample spacing at any antenna (= the lag step), CC is the rejection limit on the auto correlation (see (c)); LX is the rejection limit for peaks in the cross correlation, and CY is used in the cycling correction (see section 2, pg. 72). Since the Pet is a BASIC machine, the machine language subroutine used to communicate with the Apple must be loaded from BASIC. It is placed in Tape Buffer #2, and a listing is given in Section 5 (pg. 77).

b) Data input

The locations of the incoming and outgoing data are shown in Section 6 (pg. 78). The data (which apply to a single height) are read from these locations and put in the appropriate matrices.

c) Fast fading test and definition of one point in the mean auto correlation

ρ is calculated for increasing lag (1 to 5) until it is less than CC. If $\rho <$ CC at the first lag, the record is rejected for fast fading (i.e. noise). If $\rho \nless$ CC , its value at the last tested lag is saved for future use in the PMFCA . Otherwise linear interpolation is used to find the exact time for ρ to fall to CC. Under the Gaussian assumption this point defines the auto correlation function.

d) Limited lag

This section tries to define a more limited lag (< 16) for the peak search in the cross correlations as described by Meek (1978). Successful definition requires that all cross correlations at zero lag be greater than zero. This is checked by examining just the numerator of ρ' . At present the nominal zero lag values (not corrected for cycling) are used, otherwise interpolation to find the correlation value at true zero lag would be required.

The main purpose of this attempt is to avoid spurious peaks in the cross correlations, not to save time.

e) Peak location

A search is made for the two greatest local maxima in each cross correlation, and their index positions saved. Because of the binary nature of the correlated sequences, the possibility of exactly equal adjacent values at a peak cannot be ignored, especially if the number of blocks/record is small. The present analysis identifies the first such value, from -ve lag, as the peak. Fig. 1 illustrates

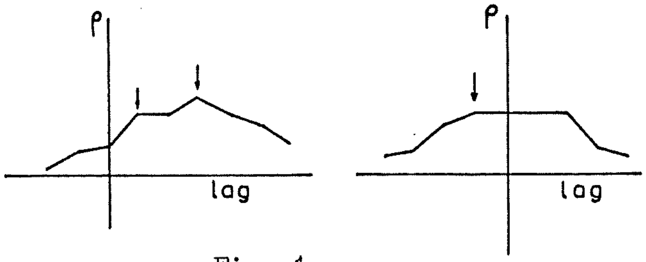


Fig. 1

two undesirable situations which could occur. The present resolution (8 blocks/record) is about 0.008 (see Appendix B, Section 5, pg. 63 for further comments) and so exactly equal values are not very likely.

f) Peak selection

Here accurate values of ρ'_{max} and t_{max} for each peak are found by a parabolic fit around the index positions found in e). Then ρ'_{max} is corrected to get ρ_{max} and the t_{max} are corrected for cycling (see section 2, pg. 71). If there is a resulting peak value less than LX it is discarded. If after this process there is a sequence without peaks, the record is rejected. The number of peaks is further reduced by comparing primary and secondary (if any) peaks in each sequence. If $\rho_{pri}/\rho_{sec} > 1.8$, then ρ_{sec} is discarded.

At this point there are only one or two peaks left in each sequence. If there is only one significant peak left in each cross correlation sequence, the NTD must be less than 0.3 for the calculation to continue; if two significant peaks in one cross correlation and one in each of the others, then one of the two combinations of peaks must have an NTD < 0.2 or the data are rejected; if two significant peaks in two or more cross correlations, then the calculation continues only if the NTD of the greatest peaks is less than 0.1 . If the data have not been rejected, the result is three time delays and three maximum correlation values.

g) Least squares fit to times

Since seven measured values are available to determine six FCA parameters, there is one degree of freedom. The derivation of the PMFCA (Section 4, pg. 73) has assumed that the NTD=0 (as it must be if the assumptions for FCA are satisfied). To make NTD=0 a weighted least squares fit (see Section 3, pg.72) of an apparent velocity vector to the time delays is made, which is then translated back into time delays. Since the peaks were chosen on the basis of low NTD, the resulting time delays will be shifted only slightly from the originals. The original ρ_{max} values are retained.

h) PMFCA

The PMFCA method is based on the Gaussian correlation assumption (e.g. Fedor,1967). Under this assumption it can be shown that the widths of the auto and all cross correlation peaks (e.g. at half the peak value) are equal (Meek,1978). Thus the width of the auto correlation theoretically defines all the widths - and is used for this purpose since it is better defined under experimental conditions than the widths of the cross correlation peaks. The remaining unknowns, for a complete definition of the correlation function (and thus the pattern parameters, velocity etc.),are the magnitudes of the three cross correlation peaks, and two of the time delays (the third being defined by NTD=0). The solution for the FCA parameters in terms of these measured values is derived in Section 4 (pg.73).

i) Output

Here the output variables are put into the 16 specified memory locations which the machine language routine sends back to the Apple. If there is no solution, the bytes reserved for FCA parameters are zeroed.

2. Cycling correction

Since all antennas are not sampled simultaneously, "0 lag" is only a nominal term in the Apple program, and a correction to the time lag (e.g. for maximum cross correlation) is required. Fig. 1 (below) shows the transfer of a particular correlation value to the Pet. The antenna cycling is in the order #1,2,3,4 at a rate of 7.5 Hz (.133 sec between antennas). Now suppose there is

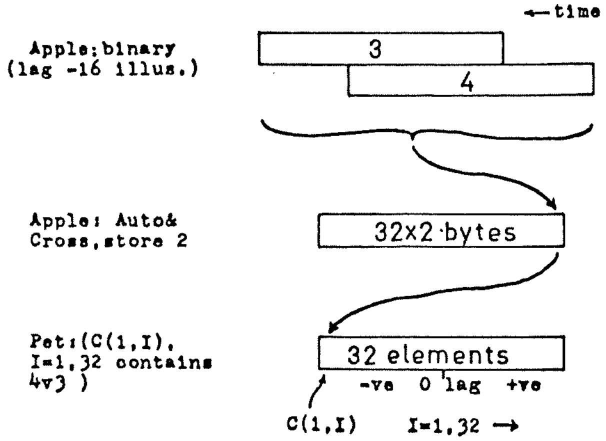

←time
Apple:binary (lag -16 illus.)
3
4
Apple: Auto& Cross,store 2
32x2 bytes
Pet:(C(1,I), I=1,32 contains 4v3)
32 elements
-ve 0 lag +ve
C(1,I)
I=1,32 →

Fig. 1

a peak in the 4v3 cross correlation exactly at zero lag (nominal). Since antenna #3 is sampled 0.133 sec before #4, this means that the 'event' must have occurred in #3 and moved to #4 in 0.133 sec (although nominally they occur simultaneously)- i.e. the pattern moves from #3 to #4. Thus if t'_{34} (the time taken for the pattern to move from #3 to #4) is the nominal value, the corrected value is:

$$t_{34} = t'_{34} + 0.133 \text{ sec} \quad ,$$

similarly: $$t_{24} = t'_{24} + 2\text{x}0.133 \text{ sec},$$

and: $$t_{14} = t'_{14} + 3\text{x}0.133 \text{ sec} \; .$$

But, by the definition of negative lag in the sketch, we are actually measuring t'_{43} etc. (not t'_{34}) - since a peak at negative lag indicates that the 'event' happened in #3 before #4. Therefore, since $t'_{43} = - t'_{34}$, the required correction is:

$$t_{43} = t'_{43} - 0.133 \text{ sec} \quad \text{etc.}$$

3. Weighted least squares fit to times

If the antenna pairs are defined by (d_i, Ψ_i), where the time delay t_i (given by the position of the maximum in the cross correlation) is taken to be positive if it indicates pattern motion in the direction Ψ_i; then the delays for a particular apparent velocity (V_a, ϕ_a) are given by:

$$t_i = \frac{d_i}{V_a} \cos(\Psi_i - \phi_a)$$

$$= \frac{\cos\phi_a}{V_a}(d_i \cos\Psi_i) + \frac{\sin\phi_a}{V_a}(d_i \sin\Psi_i) \; .$$

Let

$$\boldsymbol{\beta} = \begin{bmatrix} \dfrac{\cos\phi_a}{V_a} \\ \\ \dfrac{\sin\phi_a}{V_a} \end{bmatrix} , \quad Y= \begin{bmatrix} t_1 \\ \cdot \\ \cdot \\ \cdot \\ \circ \\ t_n \end{bmatrix} , \quad X= \begin{bmatrix} d_1\cos\psi_1 & d_1\sin\psi_1 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ d_n\cos\psi_n & d_n\sin\psi_n \end{bmatrix} .$$

and let ρ_i be the peak values of cross correlation (at delay t_i). (Usually there will be only n=3 pairs of antennas involved).Then the solution for a least squares fit of apparent velocity to time delays is (e.g. Hoel et al.,1971 Ch. 4):

$$\boldsymbol{\beta} = (X^TX)^{-1}X^TY$$

This solution minimizes the squared error in the time delays, which is

$$\sum e^2 = (X\boldsymbol{\beta} - Y)^T(X\boldsymbol{\beta} - Y).$$

If X and Y are modified according to:

$$x_{ij} = x_{ij}\rho_i$$

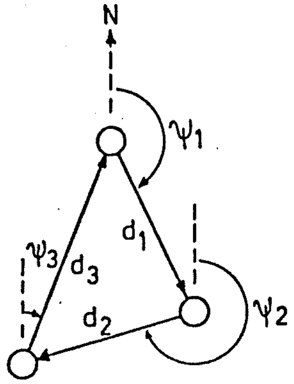
$$y_i = y_i\rho_i$$

before solving for $\boldsymbol{\beta}$, the squared error in time delays is effectively weighted by ρ_i^2 . Thus the final apparent velocity vector will agree best with the time delay corresponding to the highest cross correlation value, and this delay will be modified the least when the apparent velocity is translated back into time delays.

4. Poor man's FCA

This section derives a set of equations for determining FCA parameters from just the positions, $t_{\max_i}$, and the values of the maxima, $\rho_{\max_i}$, in the three cross correlations, and one point, (t_a, ρ_a), from the mean auto correlation. The assumptions are that the correlation function is Gaussian, and the NTD=0.

Fig. 1 describes the general antenna arrangement. The direction given to each pair defines the sign of t_{max_i}; i.e. a pattern moving in the direction of the arrow gives a positive delay.



Antenna array

Fig. 1

The auto correlation is given by:

$$\rho(0,0,t) = \exp\left(-\tfrac{1}{2}\left[V^2\left(\frac{\cos^2(\phi-\theta)}{a^2} + \frac{\sin^2(\phi-\theta)}{b^2}\right) + \frac{1}{c^2}\right]t^2\right) \qquad \ldots(1)$$

Let

$$Q = V^2\left(\frac{\cos^2(\phi-\theta)}{a^2} + \frac{\sin^2(\phi-\theta)}{b^2}\right) + \frac{1}{c^2} \quad . \qquad \ldots(2)$$

Then $\rho(0,0,t) = \exp(-\tfrac{1}{2}Qt^2)$, ...(3)

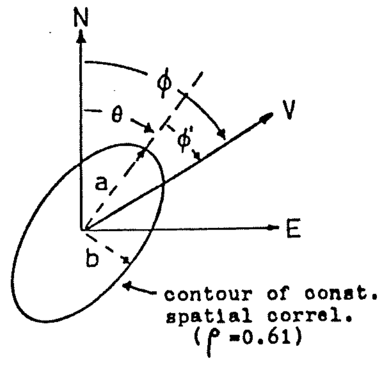
and Q can be found from one point on the auto correlation:

$$Q = \frac{-2\ln\rho_a}{t_a^2} \quad . \qquad \ldots(4)$$

The cross correlations are given by:

$$\rho(d_i,\Psi_i,t) = \exp\left[-\tfrac{1}{2}\left(\frac{(d_i\cos(\Psi_i-\theta)-V\cos(\phi-\theta)t)^2}{a^2} + \frac{(d_i\sin(\Psi_i-\theta)-V\sin(\phi-\theta)t)^2}{b^2} + \frac{t^2}{c^2}\right)\right] \qquad \ldots(5)$$

where the antenna pair and direction are defined by d_i and Ψ_i as shown in Fig. 1. The other parameters are the axes of the characteristic ellipse(a,b), the characteristic time (c), the true velocity (V), the tilt angle (θ) of the ellipse, and the drift direction



FCA parameters
Fig. 2

(ϕ). These are shown in Fig. 2.

Then for the i^{th} antenna pair, t_{max_i} is found by setting the derivative of (5) equal to zero.

$$t_{max_i} = \frac{Vd_i}{Q}\left(\frac{\cos(\phi-\theta)\cos(\psi_i-\theta)}{a^2} + \frac{\sin(\phi-\theta)\sin(\psi_i-\theta)}{b^2}\right) \qquad \ldots(6)$$

This is substituted into (4) to get ρ_{max_i} :

$$-2\ln\rho_{max_i} = d_i^2\left(\frac{\cos^2(\psi_i-\theta)}{a^2} + \frac{\sin^2(\psi_i-\theta)}{b^2}\right) - Qt^2_{max_i} \qquad \ldots(7)$$

Now define

$$m_i = \frac{-2\ln\rho_{max_i} + Qt^2_{max_i}}{d_i^2}$$

$$= \frac{1}{r^2b^2}\left(\cos^2(\psi_i-\theta) + \sin^2(\psi_i-\theta)r^2\right) \qquad \ldots(8)$$

where $r = a/b$, and let $m_{ij} = m_i - m_j$.

Then the solution for the pattern parameters is:

$$\left.\begin{aligned}
\tan(2\theta) &= \frac{m_{32}\cos2\psi_1 + m_{13}\cos2\psi_2 + m_{21}\cos2\psi_3}{-(m_{32}\sin2\psi_1 + m_{13}\sin2\psi_2 + m_{21}\sin2\psi_3)} \\
\frac{r^2}{1-r^2} &= \frac{m_1\cos^2(\psi_2-\theta) - m_2\cos^2(\psi_1-\theta)}{m_{21}} \\
b^2 &= \frac{\frac{3}{2}\left[\frac{1}{r^2}+1\right]}{m_1+m_2+m_3}, \qquad a^2 = b^2r^2
\end{aligned}\right\} \ldots(9)$$

The drift velocity is found from equation (6) for i=1,2.

$$\tan(\phi-\theta) = \frac{t_1d_2\cos(\psi_2-\theta) - t_2d_1\cos(\psi_1-\theta)}{t_2d_1\sin(\psi_1-\theta) - t_1d_2\sin(\psi_2-\theta)}\left(\frac{b^2}{a^2}\right) \qquad \ldots(10)$$

$$V = \frac{Qt_1}{d_1\left[\dfrac{\cos(\phi-\theta)\cos(\psi_1-\theta)}{a^2} + \dfrac{\sin(\phi-\theta)\sin(\psi_1-\theta)}{b^2}\right]} \qquad \ldots(11)$$

where $t_i = t_{max_i}$. Then c is found from equation (1):

$$c = \sqrt{\frac{1}{Q - V^2\left[\dfrac{\cos^2(\phi-\theta)}{a^2} + \dfrac{\sin^2(\phi-\theta)}{b^2}\right]}} \qquad \ldots(12)$$

If a^2, b^2, or c^2 is negative, the FCA solution must be rejected as physically impossible.

Table 1 gives an example for testing the programmed solution. Note that the quoted V (drift) is the pattern velocity (from eqn. 11) divided by 2.

Table 1.

antenna parameters: $d_1=d_2=d_3=156$m; $\psi_1=210^\circ$, $\psi_2=90^\circ$, $\psi_3=330^\circ$

input data	solution
$t_a=1.066$ sec	$Q = 0.53$
$\rho_a=0.74$	$a = 97.4$m
	$b = 56.0$m
	$\theta = 158^\circ$
$t_{max_i}=-2.05$s$,2.84$s$,-.79$s	$V = 16.6$m/s (drift)
	$\phi = 90^\circ$
$\rho_{max_i}=0.17,0.25,0.31$	$c = 2.2$s

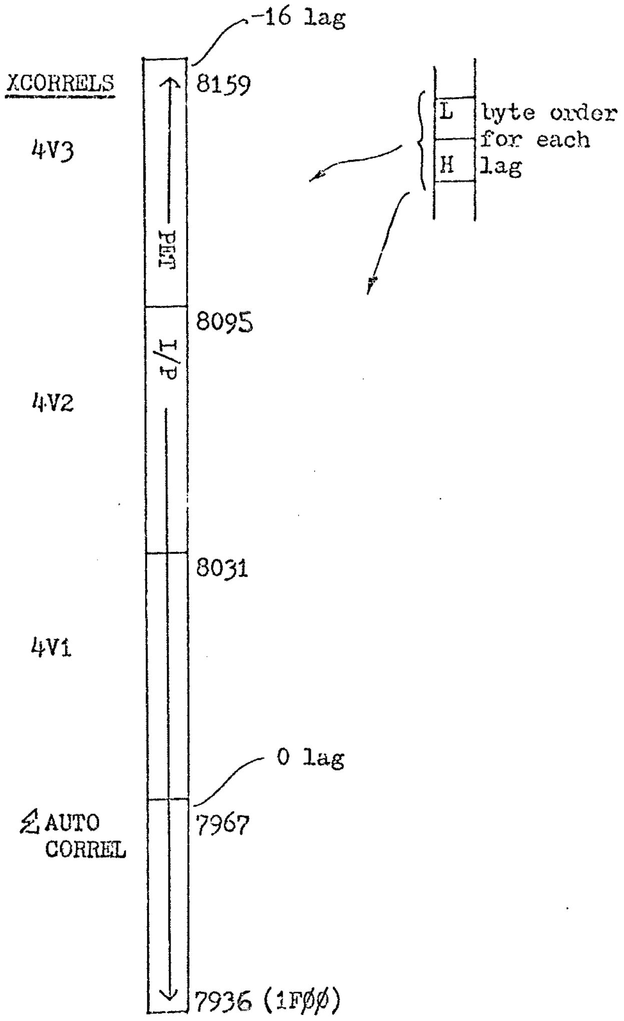
5. Pet Machine Language routine

```
ADDRESS                 PROGRAM
DEC    HEX       HEX          DEC            ASSEM.

826    $Ø33A     A9 FF        169 255        LDA #$FF
        Ø33C     8D 43 E8     141  67 232    STA $E843
        Ø33F     A2 1Ø        162  16        LDX #$1Ø
        Ø341     BD EF 1F     189 239  31    LDA $1FEF,X     OUTPUT $1Øbytes
        Ø344     8D 41 E8     141  65 232    STA $E841
        Ø347     A9 EA        169 234        LDA #$EA
        Ø349     8D 4C E8     141 76 232     STA $E84C
        Ø34C     A9 DA        169 218        LDA #$DA
        Ø34E     8D 4C E8     141  76 232    STA $E84C
        Ø351     AD 4D E8     173  77 232    LDA $E84D
        Ø354     29 Ø2         41   2        AND #$Ø2
        Ø356     FØ F9        240 249        BEQ $Ø351
        Ø358     CA           202            DEX
        Ø359     DØ E6        208 230        BNE $Ø341
        Ø35B     AD 41 E8     173  65 232    LDA $E841
        Ø35E     A9 ØØ        169   0        LDA #$ØØ
        Ø36Ø     8D 43 E8     141  67 232    STA $E843
        Ø363     A2 FØ        162 240        LDX #$FØ
        Ø365     A9 EA        169 234        LDA #$EA
        Ø367     8D 4C E8     141  76 232    STA $E84C
        Ø36A     AD 4D E8     173  77 232    LDA $E84D
        Ø36D     29 Ø2         41   2        AND #$Ø2        INPUT $FØbytes
        Ø36F     FØ F9        240 249        BEQ $Ø36A
        Ø371     Ad 41 E8     173  65 232    LDA $E841
        Ø374     9D FF 1E     157 255  30    STA $1EFF,X
        Ø377     A9 EA        169 234        LDA #$EA
        Ø379     9D 4C E8     141  76 232    STA $E84C
        Ø37C     A9 DA        169 218        LDA #$DA
        Ø37E     8D 4C E8     141  76 232    STA $E84C
        Ø381     CA           202            DEX
        Ø382     DØ E6        208 230        BNE $Ø36A
        Ø384     AD 41 E8     173  65 232    LDA $E841
903     Ø387     60            96            RTS
```

6. Input/output memory locations

VARIABLE			LOCATION
HT#		PET O/P	8191 (1FFF)
SPARE			8190
SPARE			8189
NTD			8188
REDUCED LAG			8187
V_{TRUE}	H		8186
	L		8185
ϕ_{TRUE}	H		8184
	L		8183
A	H		8182
	L		8181
B	H		8180
	L		8179
θ			8178
τ_cx10			8177
MSG			8176
D_1		PET I/P	8175 (1FEF)
D_{10}			8174
D_{100}			8173
H_1			8172
H_{10}			8171
M_1			817Ø
M_{10}			8169
NBLK			8168
DELAY			8167
GSEP			8166
CLOCKIN			8165
HT#			8164
↑			8163
SPARE			8162
			8161
↓			816Ø



7. Pet Program: version 6d

```
           10 REM-PMFCA FOR REALTIME WINDS
           20 DIMC(3,32),M(3),MX(3,2),T(3,2),R(3,2),A(6),D%(16)
           30 D=156:P=57.29578:P1=90/P:P2=210/P:P3=330/P
           40 D2=D*D:X1=D*COS(P1):X2=D*SIN(P1):X3=D*COS(P2)
           50 X4=D*SIN(P2):X5=D*COS(P3):X6=D*SIN(P3)
           60 U1=COS(2*P1):U2=COS(2*P2):U3=COS(2*P3)
           70 V1=SIN(2*P1):V2=SIN(2*P2):V3=SIN(2*P3)
           80 TS=.533:CC=.4:LX=.1:CY=.133
           90 REM N=#SAMPLES,TS=TSTEP,CC=AUTO LMT,LX=CROSS LMT
     (a)   110 DATA169,255,141,67,232,162,16,189,239,31,141,65,232,169,234,141,76
           111 DATA232,169,218,141,76,232,173,77,232,41,2,240,249,202,208,230,173,65
           112 DATA232,169,0,141,67,232,162,240,169,234,141,76,232,173,77,232,41,2,240
           113 DATA249,173,65,232,157,255,30,169,234,141,76,232,169,218,141,76,232,202
           114 DATA208,230,173,65,232,96
           120 POKE1,58:POKE2,3
           121 FORI=826TO903
           122 READX%
           123 POKEI,X%
           124 NEXTI
           130 X%=USR(X%)
           131 K=8159
           132 FORI=1TO3
           133 FORJ=1TO32
           134 C(I,J)=PEEK(K-1)*256+PEEK(K)
           135 K=K-2
           136 NEXTJ
           137 NEXTI
     (b)   138 FORI=1TO6:A(I)=(PEEK(K)+256*PEEK(K-1))/3.:K=K-2:NEXTI
           140 FORI=1TO16
           141 D%(I)=PEEK(8159+I)
           142 NEXTI
           150 DY%=D%(14)*100+D%(15)*10+D%(16)
           151 HR%=D%(13)+D%(12)*10:MN%=D%(11)      +D%(10)*10:HT%=49+3*D%(5):N=D%(9)*64
           152 IFHT%=49THENPRINT"D/T="DY%;HR%;MN%;"N=";N
           153 PRINT"HT=";HT%;"SIG";PEEK(8163);"GN=";PEEK(8162)
           180 IX=0:IR=0:IS=0:NT=255:RL=15
           181 IFA(1)=NTHENNT=254:GOTO2000
           190 A1=A(1)*A(1)/N/N:A2=A(1)/N*(1-A(1)/N)
           510 S=2
           520 FORI=1TO3
           530 X0=(C(I,17)/N-A1)/A2
           540 IFX0<STHENS=X0
           550 NEXTI
           551 X0=S
           600 RA=CC:S0=1
           610 FORI=2TO6
           630 S=SIN(1.5708*((A(I)/(N-I+1)-A1)/A2))
           640 IFS<CCTHEN690
(d)  (c)   650 S0=S
           660 NEXTI
           670 TA=(I-2)*TS:RA=S0:GOTO720
           690 IFI>2THEN710
           700 PRINT"FF":IX=7:GOTO2000
           710 TA=(I-2+(S0-CC)/(S0-S))*TS
           720 IFX0>0THEN740
           730 RL=15:GOTO750
           740 X0=SIN(1.5708*X0):RL=TA*((LOG(X0)-2)/LOG(RA))/TS+3
           741 RL=INT(RL+0.5)
           745 IFRL>15THENRL=15
```

(e)

```
750 REM-FIND PEAKS
760 M1=17-RL
770 M2=16+RL
780 FORI=1TO3
790 M=0:R(I,2)=0
800 FORJ=M1TOM2
805 S=C(I,J):IFS<N*A1THEN840
810 IFS<=C(I,J-1)THEN840
820 IFS<C(I,J+1)THEN840
830 M=M+1:IFM=1THEN836
832 IFS<=R(I,1)THEN834
833 MX(I,2)=MX(I,1):R(I,2)=R(I,1):MX(I,1)=J:R(I,1)=S:GOTO840
834 IFS>R(I,2)THENMX(I,2)=J:R(I,2)=S:GOTO840
835 GOTO840
836 MX(I,1)=J:R(I,1)=S
840 NEXTJ
850 M(I)=M
960 NEXTI
961 GOSUB1600
965 IFIX=1THENGOTO970
966 IFIX=2THENGOTO970
967 PRINT MX(1,1);MX(2,1);MX(3,1);INT(R(1,1)*100);INT(R(2,1)*100);INT(R(3,1)*100)
970 IFIR=4THENGOTO2000
```

(g)

```
1000 REM-WEIGHTED LSFIT
1010 S1=R1*R1:S2=R2*R2:S3=R3*R3
1020 Q1=X1*X1*S1+X3*X3*S2+X5*X5*S3
1030 Q=X2*X1*S1+X3*X4*S2+X5*X6*S3
1040 Q2=X2*X2*S1+X4*X4*S2+X6*X6*S3
1050 D1=Q1*Q2-Q*Q:A1=Q2/D1:A=-Q/D1:A2=Q1/D1
1060 X=T1*(X1*A1+A*X2)*S1+T2*(X3*A1+X4*A)*S2
1070 X=X+T3*(X5*A1+X6*A)*S3:Y=T1*(X1*A+X2*A2)*S1
1080 Y=Y+T2*(X3*A+X4*A2)*S2+T3*(X5*A+X6*A2)*S3
1090 Q=X*X+Y*Y
1100 GOSUB1420
1110 VA=1/SQR(Q):PA=A:REM END OF WLSFIT
1130 T1=D*COS(P1-A1)/VA:T2=D*COS(P2-A1)/VA:T3=D*COS(P3-A1)/VA
1140 VA=VA/2
```

(h)

```
1150 REM-PMFCA
1160 Q=-2*LOG(RA)/TA/TA
1170 MA=(-2*LOG(R1)+Q*T1*T1)/D2
1180 MB=(-2*LOG(R2)+Q*T2*T2)/D2
1190 MC=(-2*LOG(R3)+Q*T3*T3)/D2
1200 N2=MC-MB:N3=MA-MC:N1=MB-MA
1210 Y=N2*U1+N3*U2+N1*U3:X=-N2*V1-N3*V2-N1*V3
1220 GOSUB1420
1230 T0=A1/2:TL=A/2
1240 C1=COS(P1-T0):C2=COS(P2-T0):S1=SIN(P1-T0):S2=SIN(P2-T0)
1250 R2=(MA*C2*C2-MB*C1*C1)/N1:R2=R2/(1+R2)
1260 B2=(C1*C1/R2+S1*S1)/MA:A2=R2*B2
1265 IFA2<=0ORB2<=0THEN PRINT"IMAG AXIS":IX=5:GOTO2000
1270 AX=SQR(A2):BX=SQR(B2)
1280 X=A2*(T2*S1-T1*S2):Y=B2*(T1*C2-T2*C1)
1290 GOSUB1420
1300 PT=A+TL:PT=PT-180*(1+SGN(PT-360))
1310 Q1=COS(A1):Q2=SIN(A1)
1320 VT=Q*T1/D/(Q1*C1/A2+Q2*S1/B2)/2
1330 IFVT>0THEN1350
1340 PT=PT-180*SGN(PT-180):VT=-VT
1350 C2=1/(Q-4*VT*VT*(Q1*Q1/A2+Q2*Q2/B2))
1355 IFC2<=0THENPRINT"IMAG TC":IX=6:GOTO2000
1360 TC=SQR(C2)
1370 IFAX>BXTHEN1390
1380 A1=AX:AX=BX:BX=A1:TL=TL-90*SGN(TL-90)
1390 PRINT"FCA";INT(VT);INT(PT);INT(AX);INT(BX);INT(TL);INT(TC*10)/10
1400 REM END OF FCA
1410 GOTO2100
```

```
1420 REM ARCTAN (Y/X), 0 TO 360
1430 IFX<>0THEN1450
1440 A=90*(2-SGN(Y)):GOTO1500
1450 A=ATN(Y/X)*P+90*(2-SGN(Y)*(1+SGN(X)))
1500 A1=A/P
1510 RETURN
```

(f)

```
1600 REM SUBROUTINE WEED FOR PMFCA
1610 FORK=1TO3
1615 IFM(K)<=0THEN PRINT "NLP";K:IR=4:IX=1:RETURN
1618 IFM(K)>2THENM(K)=2
1619 MM=M(K)
1620 FORJ=1TOMM:MX=MX(K,J)
1630 Y1=C(K,MX-1):Y2=R(K,J):Y3=C(K,MX+1)
1640 A5=(Y1+Y3)/2-Y2:B5=(Y3-Y1)/2:X=-B5/A5/2
1650 S=A5*X*X+B5*X+Y2
1660 T(K,J)=(X+MX-17)*TS-K*CY
1670 R(K,J)=SIN(1.5708*((S/(N-ABS(MX-17))-A1)/A2))
1680 IFR(K,J)<LXTHENM(K)=M(K)-1
1685 IFM(K)=0THENPRINT"EXIT WEED";K:IR=4:IX=2:RETURN
1690 NEXTJ
1700 IFM(K)<2THEN1720
1710 IFR(K,1)/R(K,2)>1.8THENM(K)=M(K)-1
1720 NEXTK
1730 REM HERE M(K)=1 OR 2
1740 IS=M(1)+M(2)+M(3)-3
1750 ONIS+1GOTO1760,1800,1900,1900
1750 S1=0:S2=0:FORK=1TO3:S1=S1+T(K,1):S2=S2+ABS(T(K,1)):NEXTK
1761 NT=INT(ABS(S1)/S2*200)
1770 IFABS(S1)/S2>.3THENIR=4:IX=3:RETURN
1780 GOTO1940
1800 FORK=1TO3:IFM(K)=2THENK1=K
1805 NEXTK:IR=K1
1810 K2=3-INT(K1/3):K3=2-INT(K1/2):S1=T(K2,1)+T(K3,1)
1815 S2=ABS(T(K2,1))+ABS(T(K3,1)):NT=INT(ABS(S1+T(K1,1))/(S2+ABS(T(K1,1)))*200)
1820 FORK=1TO2:S=T(K1,K):IFABS(S1+S)/(S2+ABS(S))<.2THEN1840:NEXTK
1830 IR=4:IX=4:RETURN
1840 T(K1,1)=T(K1,K):R(K1,1)=R(K1,K):GOTO1940
1900 NT=ABS(T(1,1)+T(2,1)+T(3,1))/(ABS(T(1,1))+ABS(T(2,1))+ABS(T(3,1)))
1910 IFNT>.1THENIR=4:NT=INT(NT*200):RETURN
1920 IR=5:NT=INT(NT*200)
1940 T1=T(1,1):T2=T(2,1):T3=T(3,1):R1=R(1,1):R2=R(2,1):R3=R(3,1)
1950 RETURN
1990 REM-O/P
```

(i)

```
2000 VT=0:PT=0:AX=0:BX=0:TL=0:TC=0
2001 PRINT"ERR";NT;RL,IS;IR;IX
2100 IFVT>65280THENVT=65280
2101 X%=INT(VT/256):POKE8186,X%
2105 ER%=IS+IR*4+IX*32
2115 X%=INT(VT-X%*256):POKE8185,X%
2120 X%=INT(PT/256):POKE8184,X%:X%=INT(PT-X%*256):POKE8183,X%
2125 IFAX>65280THENAX=65280
2126 X%=INT(AX/255):POKE8182,X%
2130 X%=INT(AX-X%*256):POKE8181,X%
2135 IFBX>65280THENBX=65280
2136 X%=INT(BX/256):POKE8180,X%
2140 X%=INT(BX-X%*256):POKE8179,X%
2145 X%=INT(TL):POKE8178,X%:X%=TC*10
2146 IFX%>255THENX%=255
2147 POKE8177,X%:POKE8176,ER%
2155 POKE8188,NT:POKE8187,RL
2160 POKE8190,HR%:POKE8189,MN%:X%=D%(5):POKE8191,X%
2170 GOTO130
```

APPENDIX D

Tape output format

One tape block (592 bytes) constitutes the results of a full wind record (32 heights). It is read in 16 byte "logical records" (LR).

LR#1 : gives, in order, day#(units), day#(tens), day#(hundreds),hours(units), hours(tens), minutes(units),minutes(tens) in GMT; the number of blocks per wind record (1 block = 34 sec at present; i.e. 256 pulses at 7.5 Hz.); and three parameters which define the heights recorded. The first (DLY) gives the nominal delay before the first height gate according to: $44+5xDLY\pm1$ = program delay in μsec (assuming that the branches in the interrupt routine do not cross page boundaries)- the uncertainty is in the 'reaction time' of the Apple to an interrupt. Because of equipment design (see Chapter 3, pg. 10) the actual delay before the first height gate is D= INT((44+5xDLY±1-10)/20)x20+30 μsec (where INT means ' the integer part of'), and there is no uncertainty in the position of the first height gate. Thus the first height gate (measured with respect to the leading edge of the transmitter pulse) is at Dx(3/20)-2 Km (the 2 Km accounts for the travel time to the transmitter). The second, GSEP, defines the gate separation; ∅8 gives 3 Km (see Chapter 3, pg. 10). The third, CLKIN, defines a delay which must allow the clocking in of exactly 64 height gates (i.e. it depends on the gate separation): delay = 24+CLKINx63 μsec. This is best set by counting the gates on an oscilloscope. The last five bytes are free, but they are only accessed from the Apple (before any winds are calculated) and in this sense are not very useful. In future they will be used to feed more information to the Pet.

LR#2,3 : gives the average of the average signal in each block for the lowest to highest height gate.

LR#4,5 : gives the gain setting (0=lowest gain, to 7= highest gain) for each height gate (lowest to highest).

LR#6-37 : gives the wind results from each height (#0 to #31; referred to in some places as #1 to #32). These are the direct output of the Pet. The present output in each LR is as follows:

Byte#1 : height number (0-31). This is essentially a free byte since the heights are always in the same order, however it has been used to advantage in reconstructing data recorded on a less than perfect tape machine.

Byte#2,3 : hour and minute. These are also redundant, and may not be relied upon in future modifications.

Byte#4 : If this is 0-200 it is the value of the NTDx200 (for the largest peaks in the cross correlations). If it is 254, the record was rejected due to constant input signal. If it is 255, the NTD was not calculated either because the record was rejected for fast fading, or at least one cross correlation did not have peaks of sufficient size (i.e.$>$0.1) to define a time delay.

Byte#5 : is the reduced lag used for finding peaks- a value of 16 (now 15) means that no such reduction was possible.

Byte#6,7 : gives the true drift velocity (B_6x256+B_7) in m/s. Since a true velocity > 256m/s is very unlikely (and probably spurious if it does occur), #6 could be regarded as a free byte, or could be used to increase the accuracy of the output value.

Byte#8,9 : gives the true drift direction in degrees (0-360) East of North;
$\phi = B_8 \text{x} 256 + B_9$ degrees.

Byte#10,11 : gives the major axis of the characteristic ellipse (at ρ=0.61) on the ground ; A = B_{10}x256+B_{11} metres.

Byte#12,13 : gives the minor axis of the characteristic ellipse; B=B_{12}x256+B_{13}.

Byte#14 : gives the tilt angle of the major axis of the characteristic ellipse in degrees East of North (0-180).

Byte#15 : gives the characteristic time (at ρ=0.61) in seconds times 10. (a value $\geqslant$ 256 is set equal to 256).

Byte#16 : is a message which indicates reasons for rejection of data - or the process by which data was accepted. It is the sum of three message indicators at present; IER=IS+IRx4+IXx32 (IS=0-3, IR=0-5, IX=0-7), and there is some redundancy which may be cleared up a a later date. The important configurations are listed:

IX=7: data rejected for fast fading
IX=1,2 : No local peaks > 0.1 in at least one cross correlation
IX=3,4 or IX=0 and IR=4 : NTD too high so data rejected
IR=5 : two or more peaks in two or more cross correlations but NTD < 0.1 so FCA attempted.
IX=5,6 : FCA attempted, but no acceptable solution found.
IX=0 and NTD≠254 : FCA solution found.

Note that if no acceptable FCA solution has been found, bytes#6-15 are zero.

Pg.85 shows a dump of a tape block.

Dump of one tape block(HEX)

```
LR#
 1   04000102  02000510  39081400  00000000
 2   05040404  04082129  1F353737  31231D18
 3   15191111  28888C41  15100A04  03030304
 4   07070707  07070707  07070706  02020204
 5   05060603  01010000  01010406  06060606
 6   001632FF  10000000  00000000  000000E0
 7   011632FF  10000000  00000000  000000E0
 8   021632FF  10000000  00000000  000000E0
 9   031632FF  10000000  00000000  000000E0
10   041632FF  10000000  00000000  000000E0
11   051632FF  10000000  00000000  000000E0
12   06163208  10000A00  5F007600  4DB12A17
13   07163217  10000A00  51006500  30042B0D
14   0816326B  10000000  00000000  00000091
15   091632FF  10000000  00000000  00000050
16   0A1632C8  10000000  00000000  00000012
17   0B163272  10000000  00000000  00000013
18   0C16321D  10000000  00000000  00000012
19   0D163202  10002D01  13008200  40030D00
20   0E16320C  10002D01  23006900  3BB20B05
21   0F163279  10000000  00000000  00000091
22   101632FF  10000000  00000000  000000E0
23   111632FF  10000000  00000000  000000E0
24   121632FF  10000000  00000000  000000E0
25   131632FF  10000000  00000000  000000E0
26   1416320C  10001800  62009F00  70081B09
27   15163200  10001100  55008800  6E0F1900
28   16163207  10001000  62008C00  5E0B1700
29   17163210  10001800  5900A500  51001400
30   1816321F  10001700  4F005E00  37181200
31   19163289  00000000  00000000  00000091
32   1A1632FF  10000000  00000000  00000050
33   1B1632FF  10000000  00000000  000000E0
34   1C1632FF  10000000  00000000  000000E0
35   1D1632FF  10000000  00000000  000000E0
36   1E1632FF  10000000  00000000  000000E0
37   1F1632FF  10000000  00000000  000000E0
```

#blocks/rec (is now Ø8)

day#(104) hr(22) min(50)

DLY

GSEP

CLKIN

Avg. sig. (ht#0-31)

Gain setting (ht# 0-31)

ht#

hr

min

MSG

τ_cx10

tilt of A

B

A

NTDx200

Reduced lag

V(true) (10 m/s)

ϕ (true) (95^{o}EofN)

APPENDIX E

Machine language reference

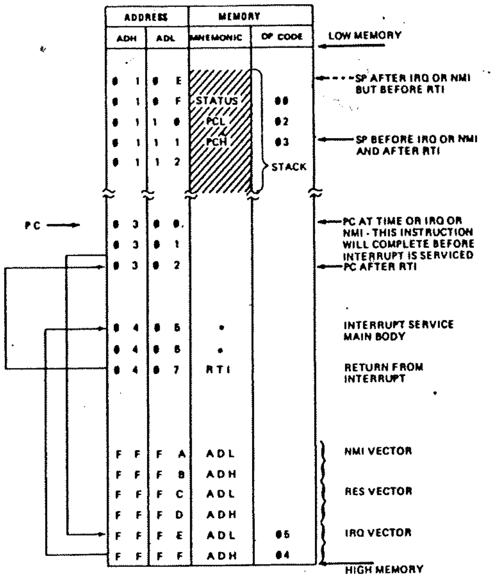


FIG. 1 IRQ, NMI, RTI, BRK OPERATION

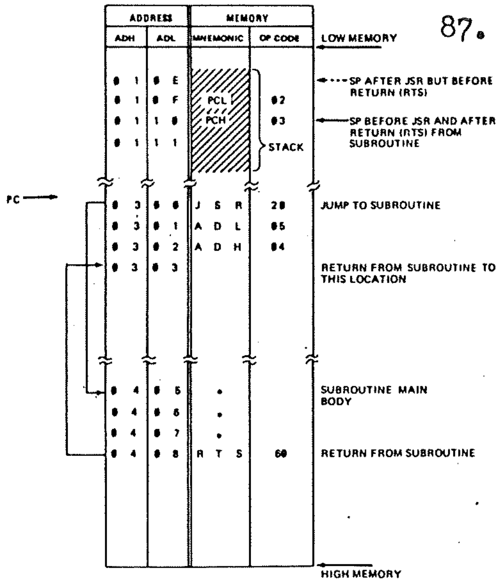


FIG. 2 JSR, RTS OPERATION

ASSEMBLER DIRECTIVES

- .OPT - IF USED MUST BE THE FIRST EXECUTABLE STATEMENT IN THE PROGRAM.
- .OPTIONS ARE;- (OPTIONS LISTED ARE THE DEFAULT VALUE. TURNED OFF BY (NO) PREFIX.)

  COUNT (COU OR CNT) - LIST ALL INSTRUCTIONS AND THEIR USAGE.
  NOGENERATE (NOG) - DO NOT GENERATE MORE THAN ONE LINE OF CODE FOR ASCII STRINGS.
  XREF (XRE) - PRODUCE A CROSS-REFERENCE LIST IN THE SYMBOL TABLE.
  ERRORS (ERR) - CREATE AN ERROR FILE.
  MEMORY (MEM) - CREATE AN ASSEMBLER OBJECT OUTPUT FILE.
  LIST (LIS) - PRODUCE A FULL ASSEMBLY LISTING.
- .BYTE - PRODUCES A SINGLE BYTE IN MEMORY EQUAL TO EACH OPERAND SPECIFIED.
- .WORD - PRODUCES TWO BYTES IN MEMORY EQUAL TO EACH OPERAND SPECIFIED.

*= - DEFINES THE BEGINNING OF A NEW PROGRAM COUNTER SEQUENCE.

- .PAGE - ADVANCES THE LISTING TO THE TOP OF A NEW PAGE.
- .END - DEFINES THE END OF A SOURCE PROGRAM.

LABELS:

LABELS BEGIN IN COLUMN 1 AND ARE SEPARATED FROM THE INSTRUCTION BY AT LEAST ONE SPACE.
LABELS CAN BE UP TO 6 ALPHANUMENIC CHARACTERS LONG AND MUST BEGIN WITH AN ALPHA CHARACTER.
A, X, Y, S, AND P ARE RESERVED AND CANNOT BE USED AS LABELS.

LABEL = EXPRESSION CAN BE USED TO EQUATE LABELS TO INSTRUCTIONS.

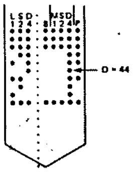
LABEL *= * + N CAN BE USED TO RESERVE AREAS IN MEMORY

CHARACTERS USED AS SPECIAL PREFIXES:

.	INDICATES AN ASSEMBLER DIRECTIVE
#	SPECIFIES THE IMMEDIATE MODE OF ADDRESSING.
$	SPECIFIES A HEXADECIMAL CHARACTER.
@	SPECIFIES AN OCTAL NUMBER.
%	SPECIFIES A BINARY NUMBER.
'	SPECIFIES AN ASCII LITERAL CHARACTER.
( )	INDICATES INDIRECT ADDRESSING.
;	IN COLUMN 1 INDICATES A COMMENT.

ASCII CHARACTER SET (7-BIT CODE)

LSD \ MSD		Ø ØØØ	1 ØØ1	2 Ø1Ø	3 Ø11	4 1ØØ	5 1Ø1	6 11Ø	7 111
Ø	ØØØØ	NUL	DLE	SP	Ø	@	P		p
1	ØØØ1	SOH	DC1	!	1	A	Q	a	q
2	ØØ1Ø	STX	DC2	"	2	B	R	b	r
3	ØØ11	ETX	DC3	#	3	C	S	c	s
4	Ø1ØØ	EOT	DC4	$	4	D	T	d	t
5	Ø1Ø1	ENG	NAK	%	5	E	U	e	u
6	Ø11Ø	ACK	SYN	&	6	F	V	f	v
7	Ø111	BEL	ETB	'	7	G	W	g	w
8	1ØØØ	BS	CAN	(	8	H	X	h	x
9	1ØØ1	HT	EM	)	9	I	Y	i	y
A	1Ø1Ø	LF	SUB	*	:	J	Z	j	z
B	1Ø11	VT	ESC	+	;	K	[	k	
C	11ØØ	FF	FS	,	<	L	\	l	|
D	11Ø1	CR	GS	–	=	M	]	m	
E	111Ø	SO	RS	.	>	N	↑	n	~
F	1111	SI	VS	/	?	□	←	o	DEL



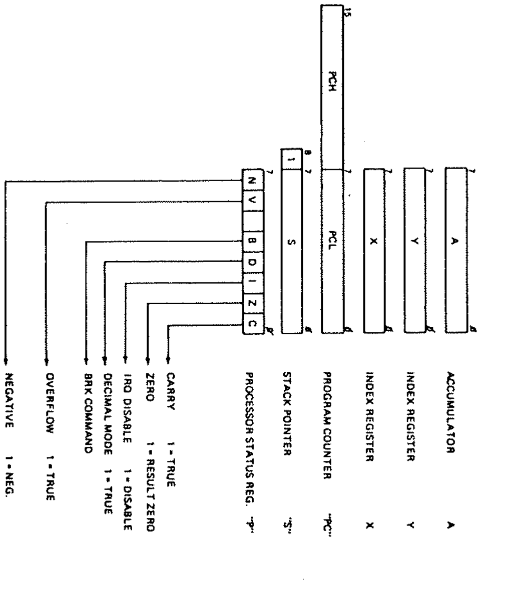
MOS TECHNOLOGY, INC.

Valley Forge Corporate Center
950 Rittenhouse Road
Norristown, Pa. 19401

MCS650X INSTRUCTION SET SUMMARY

(6501 Thru 6506)

PROCESSOR PROGRAMMING MODEL



INSTRUCTIONS		IMMEDIATE			ABSOLUTE			ZERO PAGE			ACCUM			IMPLIED			(IND,X)			(IND),Y			Z,PAGE,X			ABS,X			ABS,Y			RELATIVE			INDIRECT			Z,PAGE,Y			CONDITION CODES					
MNEMONIC	OPERATION	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	N	Z	C	I	D	V
ADC	A+M+C → A (1)	69	2	2	6D	4	3	65	3	2							61	6	2	71	5	2	75	4	2	7D	4	3	79	4	3										✓	✓	✓	-	-	✓
AND	A∧M → A (1)	29	2	2	2D	4	3	25	3	2							21	6	2	31	5	2	35	4	2	3D	4	3	39	4	3										✓	✓	-	-	-	-
ASL	C ← [7 … 0] ← 0				0E	6	3	06	5	2	0A	2	1										16	6	2	1E	7	3													✓	✓	✓	-	-	-
BCC	BRANCH ON C=0 (2)																															90	2	2							-	-	-	-	-	-
BCS	BRANCH ON C=1 (2)																															B0	2	2							-	-	-	-	-	-
BEQ	BRANCH ON Z=1 (2)																															F0	2	2							-	-	-	-	-	-
BIT	A∧M				2C	4	3	24	3	2																															M_7	✓	-	-	-	M_6
BMI	BRANCH ON N=1 (2)																															30	2	2							-	-	-	-	-	-
BNE	BRANCH ON Z=0 (2)																															D0	2	2							-	-	-	-	-	-
BPL	BRANCH ON N=0 (2)																															10	2	2							-	-	-	-	-	-
BRK	(See Fig. 1)													00	7	1																									-	-	-	1	-	-
BVC	BRANCH ON V=0 (2)																															50	2	2							-	-	-	-	-	-
BVS	BRANCH ON V=1 (2)																															70	2	2							-	-	-	-	-	-
CLC	0 → C													18	2	1																									-	-	0	-	-	-
CLD	0 → D													D8	2	1																									-	-	-	-	0	-
CLI	0 → I													58	2	1																									-	-	-	0	-	-
CLV	0 → V													B8	2	1																									-	-	-	-	-	0
CMP	A-M (1)	C9	2	2	CD	4	3	C5	3	2							C1	6	2	D1	5	2	D5	4	2	DD	4	3	D9	4	3										✓	✓	✓	-	-	-
CPX	X-M	E0	2	2	EC	4	3	E4	3	2																															✓	✓	✓	-	-	-
CPY	Y-M	C0	2	2	CC	4	3	C4	3	2																															✓	✓	✓	-	-	-
DEC	M-1 → M				CE	6	3	C6	5	2													D6	6	2	DE	7	2													✓	✓	-	-	-	-
DEX	X-1 → X													CA	2	1																									✓	✓	-	-	-	-
DEY	Y-1 → Y													88	2	1																									✓	✓	-	-	-	-
EOR	A⊻M → A (1)	49	2	2	4D	4	3	45	3	2							41	6	2	51	5	2	55	4	2	5D	4	3	59	4	3										✓	✓	-	-	-	-
INC	M+1 → M				EE	6	3	E6	5	2													F6	6	2	FE	7	3													✓	✓	-	-		
INX	X+1 → X													E8	2	1																									✓	✓	-	-	-	-
INY	Y+1 → Y													C8	2	1																									✓	✓	-	-	-	-
JMP	JUMP TO NEW LOC				4C	3	3																												6C	5	3				-	-	-	-	-	-
JSR	(See Fig. 2) JUMP SUB				20	6	3																																		-	-	-	-	-	-
LDA	M → A (1)	A9	2	2	AD	4	3	A5	3	2							A1	6	2	B1	5	2	B5	4	2	BD	4	3	B9	4	3										✓	✓	-	-	-	-

MNEMONIC	OPERATION	IMMEDIATE			ABSOLUTE			ZERO PAGE			ACCUM.			IMPLIED			(IND,X)			(IND),Y			Z,PAGE,X			ABS,X			ABS,Y			RELATIVE			INDIRECT			Z,PAGE,Y			CONDITION CODES					
		OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	N	Z	C	I	D	V
LDX	M → X (1)	A2	2	2	AE	4	3	A6	3	2																			BE	4	3							B6	4	2	✓	✓	-	-	-	-
LDY	M → Y (1)	A0	2	2	AC	4	3	A4	3	2													B4	4	2	BC	4	3													✓	✓	-	-	-	-
LSR	0 → [7 … 0] → C				4E	6	3	46	5	2	4A	2	1										56	6	2	5E	7	3													0	✓	✓	-	-	-
NOP	NO OPERATION													EA	2	1																									-	-	-	-	-	-
ORA	A∨M → A	09	2	2	0D	4	3	05	3	2							01	6	2	11	5	2	15	4	2	1D	4	3	19	4	3										✓	✓	-			-
PHA	A → Ms S-1 → S													48	3	1																									-	-	-	-	-	-
PHP	P → Ms S-1 → S													08	3	1																									-	-	-	-	-	-
PLA	S+1 → S Ms → A													68	4	1																									✓	✓	-	-		-
PLP	S+1 → S Ms → P													28	4	1																									(RESTORED)					
ROL	← [7 … 0] ← [C] ←				2E	6	3	26	5	2	2A	2	1										36	6	2	3E	7	3													✓	✓	✓	-		
RTI	(See Fig. 1) RTRN INT													40	6	1																									(RESTORED)					
RTS	(See Fig. 2) RTRN SUB													60	6	1																									-	-	-	-	-	-
SBC	A-M-C → A (1)	E9	2	2	ED	4	3	E5	3	2							E1	6	2	F1	5	2	F5	4	2	FD	4	3	F9	4	3										✓	✓	(3)			✓
SEC	1 → C													38	2	1																									-	-	1	-	-	-
SED	1 → D													F8	2	1																									-	-	-	-	1	-
SEI	1 → I													78	2	1																									-	-		1		
STA	A → M				8D	4	3	85	3	2							81	6	2	91	6	2	95	4	2	9D	5	3	99	5	3										-	-	-	-	-	-
STX	X → M				8E	4	3	86	3	2																												96	4	2	-	-	-	-	-	-
STY	Y → M				8C	4	3	84	3	2													94	4	2																-	-	-	-	-	-
TAX	A → X													AA	2	1																									✓	✓				
TAY	A → Y													A8	2	1																									✓	✓	-			
TSX	S → X													BA	2	1																									✓	✓	-	-	-	-
TXA	X → A													8A	2	1																									✓	✓		-		
TXS	X → S													9A	2	1																									-	-	-	-	-	-
TYA	Y → A													98	2	1																									✓	✓	-	-	-	-

(1) ADD 1 TO "N" IF PAGE BOUNDRY IS CROSSED

(2) ADD 1 TO "N" IF BRANCH OCCURS TO SAME PAGE
ADD 2 TO "N" IF BRANCH OCCURS TO DIFFERENT PAGE

(3) CARRY NOT = BORROW

X INDEX . X
Y INDEX Y
A ACCUMULATOR
M MEMORY PER EFFECTIVE ADDRESS
Ms MEMORY PER STACK POINTER

+ ADD
- SUBTRACT
∧ AND
∨ OR

⊻ EXCLUSIVE OR
✓ MODIFIED
- NOT MODIFIED
M_7 MEMORY BIT 7
M_6 MEMORY BIT 6

N NO. CYCLES
# NO. BYTES

OP - CODE TABLE

LSD / MSD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	LSD / MSD
0	BRK	ORA-IND, X				ORA Z, Page	ASL-Z,PAGE		PHP	ORA-IMM	ASL A			ORA ABS	ASL-ABS		0
1	BPL	ORA-IND, Y				ORA-Z, Page,X	ASL-Z,Page,X		CLC	ORA-ABS,Y				ORA-ABS,X	ASL-ABS, X		1
2	JSR	AND-IND,X			BIT-Z,Page	AND Z, Page	ROL-Z, Page		PLP	AND-IMM	ROL A		BIT ABS	AND ABS	ROL-ABS		2
3	BMI	AND-IND,Y				AND-Z,Page, X	ROL-Z,Page,X		SEC	AND-ABS,Y				AND ABS,X	ROL-ABS,X		3
4	RTI	EOR-IND,X				EOR Z, Page	LSR-Z,Page		PHA	EOR-IMM	LSR-A		JMP-ABS	EOR ABS	LSR-ABS		4
5	BVC	EOR-IND,Y				EOR-Z,Page,X	LSR-Z,Page,X		CLI	EOR-ABS,Y				EOR-ABS, X	LSR-ABS, X		5
6	RTS	ADC-IND, X				ADC-Z, Page			PLA	ADC-IMM			JMP IND	ADC-ABS			6
7	BVS	ADC-IND, Y				ADC-Z,PageX			SEI	ADC-ABS,Y				ADC ABS, X			7
8		STA-IND, X			STY-Z, Page	STA-Z, Page	STX-Z Page		DEY		TXA		STY-ABS	STA-ABS	STX-ABS		8
9	BCC	STA-IND, Y			STY-Z,Page,X	STA-Z,Page, X	STX-Z,Page,Y		TYA	STA-ABS, Y	TXS			STA-ABS, X			9
A	LDY-IMM	LDA-IND, X	LDX-IMM		LDY-Z Page	LDA-Z Page	LDX-Z Page		TAY	LDA IMM	TAX		LDY ABS	LDA ABS	LDX ABS		A
B	BCS	LDA-IND, Y			LDY-Z, PageX	LDA-Z,Page,X	LDX-Z, PageY		CLV	LDA-ABS, Y	TSX		LDY-ABS, X	LDA-ABS, X	LDX-ABS, Y		B
C	CPY-IMM	CMP-IND, X			CPY-Z Page	CMP-Z Page	DEC-Z Page		INY	CMP-IMM	DEX		CPY ABS	CMP ABS	DEC ABS		C
D	BNE	CMP-IND, Y				CMP-Z,Page,X	DEC-Z,Page,X		CLD	CMP-ABS, Y				CMP-ABS, X	DEC-ABS, X		D
E	CPX-IMM	SBC-IND, X			CPX-Z Page	SBC-Z Page	INC-Z Page		INX	SBC IMM	NOP		CPX ABS	SBC ABS	INC-ABS		E
F	BEQ	SBC-IND, Y				SBC-Z,Page,X	INC-Z,Page,X		SED	SBC-ABS, Y				SBC-ABS, X	INC-ABS, X		F

IMM - IMMEDIATE ADDRESSING – THE OPERAND IS CONTAINED IN THE SECOND BYTE OF THE INSTRUCTION

ABS - ABSOLUTE ADDRESSING – THE SECOND BYTE OF THE INSTRUCTION CONTAINS THE 8 LOW ORDER BITS OF THE EFFECTIVE ADDRESS. THE THIRD BYTE CONTAINS THE 8 HIGH ORDER BITS OF THE EFFECTIVE ADDRESS.

Z. PAGE- ZERO PAGE ADDRESSING – SECOND BYTE CONTAINS THE 8 LOW ORDER BITS OF THE EFFECTIVE ADDRESS. THE 8 HIGH ORDER BITS ARE ZERO.

A. - ACCUMULATOR – ONE BYTE INSTRUCTION OPERATING ON THE ACCUMULATOR

Z. PAGE,X - Z. PAGE,Y ZERO PAGE INDEXED - THE SECOND BYTE OF THE INSTRUCTION IS ADDED TO THE INDEX (CARRY IS DROPPED) TO FORM THE LOW ORDER BYTE OF THE EA. THE HIGH ORDER BYTE OF THE EA IS ZEROS

ABS,X ABS,Y ABSOLUTE INDEXED THE EFFECTIVE ADDRESS IS FORMED BY ADDING THE INDEX TO THE SECOND AND THIRD BYTE OF THE INSTRUCTION

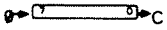
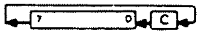
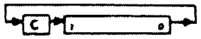
(IND, X) - INDEXED INDIRECT – THE SECOND BYTE OF THE INSTRUCTION IS ADDED TO THE X INDEX, DISCARDING THE CARRY. THE RESULTS POINTS TO A LOCATION ON PAGE ZERO WHICH CONTAINS THE 8 LOW ORDER BITS OF THE EA, THE NEXT BYTE CONTAINS THE 8 HIGH ORDER BITS

(IND),Y INDIRECT INDEXED – THE SECOND BYTE OF THE INSTRUCTION POINTS TO A LOCATION IN PAGE ZERO. THE CONTENTS OF THIS MEMORY LOCATION IS ADDED TO THE Y INDEX, THE RESULT BEING THE LOW ORDER EIGHT BITS OF THE EA. THE CARRY FROM THIS OPERATION IS ADDED TO THE CONTENTS OF THE NEXT PAGE ZERO LOCATION, THE RESULT BEING THE 8 HIGH ORDER BITS OF THE EA

INSTRUCTION SET – OP CODES, Execution Time, Memory Requirements

INSTRUCTIONS		IMMEDIATE			ABSOLUTE			ZERO PAGE			ACCUM.			IMPLIED			(IND,X)			(IND),Y			Ƶ,PAGE,X			ABS,X			ABS,Y			RELATIVE			INDIRECT			Ƶ,PAGE,Y			CONDITION CODES					
MNEMONIC	OPERATION	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	N	Ƶ	C	I	D	V
A D C	A+M+C → A (4) (1)	69	2	2	6D	4	3	65	3	2							61	6	2	71	5	2	75	4	2	7D	4	3	79	4	3										✓	✓	✓	–	–	✓
A N D	A∧M → A (1)	29	2	2	2D	4	3	25	3	2							21	6	2	31	5	2	35	4	2	3D	4	3	39	4	3										✓	✓	–	–	–	–
A S L	C ← [7 0] ← Ø				ØE	6	3	Ø6	5	2	ØA	2	1										16	6	2	1E	7	3													✓	✓	✓	–	–	–
B C C	BRANCH ON C=Ø (2)																															9Ø	2	2							–	–	–	–	–	–
B C S	BRANCH ON C=1 (2)																															BØ	2	2							–	–	–	–	–	–
B E Q	BRANCH ON Ƶ=1 (2)																															FØ	2	2							–	–	–	–	–	–
B I T	A∧M				2C	4	3	24	3	2																															M_7	✓	–	–	–	M_6
B M I	BRANCH ON N=1 (2)																															3Ø	2	2							–	–	–	–	–	–
B N E	BRANCH ON Ƶ=Ø (2)																															DØ	2	2							–	–	–	–	–	–
B P L	BRANCH ON N=Ø (2)																															1Ø	2	2							–	–	–	–	–	–
B R K	(See Fig. 1)													ØØ	7	1																									–	–	–	1	–	–
B V C	BRANCH ON V=Ø (2)																															5Ø	2	2							–	–	–	–	–	–
B V S	BRANCH ON V=1 (2)																															7Ø	2	2							–	–	–	–	–	–
C L C	Ø → C													18	2	1																									–	–	Ø	–	–	–
C L D	Ø → D													D8	2	1																									–	–	–	–	Ø	–
C L I	Ø → I													58	2	1																									–	–	–	Ø	–	–
C L V	Ø → V													B8	2	1																									–	–	–	–	–	Ø
C M P	A–M (1)	C9	2	2	CD	4	3	C5	3	2							C1	6	2	D1	5	2	D5	4	2	DD	4	3	D9	4	3										✓	✓	✓	–	–	–
C P X	X–M	EØ	2	2	EC	4	3	E4	3	2																															✓	✓	✓	–	–	–
C P Y	Y–M	CØ	2	2	CC	4	3	C4	3	2																															✓	✓	✓	–	–	–
D E C	M–1 → M				CE	6	3	C6	5	2													D6	6	2	DE	7	3													✓	✓	–	–	–	–
D E X	X–1 → X													CA																											✓	✓	–	–	–	–
D E Y	Y–1 → Y													88																											✓	✓	–	–	–	–
E O R	A ⊻ M → A (1)	49	2	2	4D	4	3	45	3	2							41	6	2	51	5	2	55	4	2	5D	4	3	59	4	3										✓	✓	–	–	–	–
I N C	M+1 → M				EE	6	3	E6	5	2													F6	6	2	FE	7	3													✓	✓	–	–	–	–
I N X	X+1 → X													E8	2	1																									✓	✓	–	–	–	–
I N Y	Y+1 → Y													C8	2	1																									✓	✓	–	–	–	–
J M P	JUMP TO NEW LOC.				4C	3	3																												6C	5	3				–	–	–	–	–	–
J S R	(See Fig. 2) JUMP SUB				2Ø	6	3																																		–	–	–	–	–	–
L D A	M → A (1)	A9	2	2	AD	4	3	A5	3	2							A1	6	2	B1	5	2	B5	4	2	BD	4	3	B9	4	3										✓	✓	–	–	–	–

CMP: C=1 if M≤A
C=0 if M>A

		IMMEDIATE			ABSOLUTE			ZERO PAGE			ACCUM.			IMPLIED			(IND,X)			(IND),Y			Z, PAGE, X			ABS, X			ABS, Y			RELATIVE			INDIRECT			Z, PAGE, Y			CONDITION CODES					
MNEMONIC	OPERATION	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	N	Z	C	I	D	V
L D X	M → X (1)	A2	2	2	AE	4	3	A6	3	2																			BE	4	3							B6	4	2	✓	✓	–	–	–	–
L D Y	M → Y (1)	A0	2	2	AC	4	3	A4	3	2													B4	4	2	BC	4	3													✓	✓	–	–	–	–
L S R					4E	6	3	46	5	2	4A	2	1										56	6	2	5E	7	3													0	✓	✓	–	–	–
N O P	NO OPERATION													EA	2	1																									–	–	–	–	–	–
O R A	A V M → A	09	2	2	0D	4	3	05	3	2							01	6	2	11	5	2	15	4	2	1D	4	3	19	4	3										✓	✓	–	–	–	–
P H A	A → Ms S-1 → S													48	3	1																									–	–	–	–	–	–
P H P	P → Ms S-1 → S													08	3	1																									–	–	–	–	–	–
P L A	S+1 → S Ms → A													68	4	1																									✓	✓	–	–	–	–
P L P	S+1 → S Ms → P													28	4	1																									(RESTORED)					
R O L					2E	6	3	26	5	2	2A	2	1										36	6	2	3E	7	3													✓	✓	✓	–	–	–
R O R					6E	6	3	66	5	2	6A	2	1										76	6	2	7E	7	3													✓	✓	✓	–	–	–
R T I	(See Fig. 1) RTRN. INT.													40	6	1																									(RESTORED)					
R T S	(See Fig. 2) RTRN SUB													60	6	1																									–	–	–	–	–	–
S B C	A-M-$\bar{C}$ → A (1)	E9	2	2	ED	4	3	E5	3	2							E1	6	2	F1	5	2	F5	4	2	FD	4	3	F9	4	3										✓	✓	(3)	–	–	✓
S E C	1 → C													38	2	1																									–	–	1	–	–	–
S E D	1 → D													F8	2	1																									–	–	–	–	1	–
S E I	1 → I													78	2	1																									–	–	–	1	–	–
S T A	A → M				8D	4	3	85	3	2							81	6	2	91	6	2	95	4	2	9D	5	3	99	5	3										–	–	–			
S T X	X → M				8E	4	3	86	3	2																												96	4	2	–	–	–	–	–	–
S T Y	Y → M				8C	4	3	84	3	2													94	4	2																–	–	–	–	–	–
T A X	A → X													AA	2	1																									✓	✓	–	–	–	–
T A Y	A → Y													A8	2	1																									✓	✓	–			–
T S X	S → X													BA	2	1																									✓	✓	–	–	–	
T X A	X → A													8A	2	1																									✓	✓		–		–
T X S	X → S													9A	2	1																									–	–		–		
T Y A	Y → A													98	2	1																									✓	✓	–	–	–	–

(1) ADD 1 TO "N" IF PAGE BOUNDRY IS CROSSED

(2) ADD 1 TO "N" IF BRANCH OCCURS TO SAME PAGE.
ADD 2 TO "N" IF BRANCH OCCURS TO DIFFERENT PAGE.

(3) CARRY NOT = BORROW.

(4) IF IN DECIMAL MODE Z FLAG IS INVALID.
ACCUMULATOR MUST BE CHECKED FOR ZERO RESULT.

X INDEX . X
Y INDEX Y
A ACCUMULATOR
M MEMORY PER EFFECTIVE ADDRESS
Ms MEMORY PER STACK POINTER

+ ADD
– SUBTRACT
Λ AND
V OR

⊻ EXCLUSIVE OR
✓ MODIFIED
- NOT MODIFIED
M_7 MEMORY BIT 7
M_6 MEMORY BIT 6

N NO. CYCLES
NO. BYTES