(NASA-CR-14113) COMPUTER PROGRAM CORDET (Ohio Univ.) 23 p HC $3.25 CSCL 09B

N75-13547

Unclas

G3/61 05576

TECHNICAL MEMORANDUM (NASA) 14

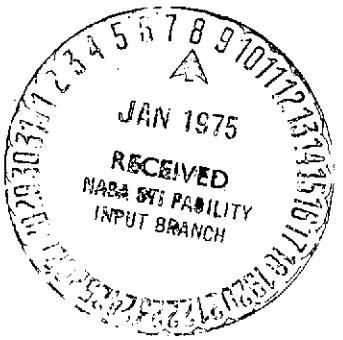
COMPUTER PROGRAM CORDET

A simulation tool is described for use in the design and analysis of digital phase-locked loops, with specific application to the DPLL in the Ohio University Omega receiver base.

by

R. A. Palkovic
Avionics Engineering Center
Department of Electrical Engineering
Ohio University
Athens, Ohio 45701

November, 1974



Supported by

National Aeronautics and Space Administration
Langley Research Center
Langley Field, Virginia
Grant NGR 36-009-017

TABLE OF CONTENTS

LIST OF FIGURES

I ABSTRACT

A Fortran IV computer program provides convenient simulation of an all-digital phase-lock loop (DPLL). The DPLL functions as the primary phase processor for the Ohio University Omega prototype navigation receiver.

This project has been supported by NASA Langley Research Center Grant NGR 36-009-017 for application to general aviation VLF receiver systems.

II INTRODUCTION

This paper describes a Fortran IV simulation study of the all-digital phase-lock loop (DPLL) which was designed by the NASA Omega staff of the Ohio University Avionics Engineering Center. The DPLL forms the heart of the Omega navigation receiver prototype. Through the DPLL, the phase of the 10.2 KHz Omega signal is estimated when the true signal phase is contaminated with noise. The study has provided a convenient means of evaluating loop performance in a variety of noise environments, and has proved to be a useful tool for evaluating design changes.

The goals of the simulation were threefold:

A. To analyze the circuit on a bit-by-bit level in order to evaluate the overall design;

B. To see easily the effects of proposed design changes prior to actual breadboarding; and

C. To determine the optimum integration time for the DPLL in an environment typical of general aviation conditions.

III DESCRIPTION OF THE SIMULATION

The digital nature of the circuit indicated the use of the digital computer as the most convenient simulation tool. Since the circuit was already built and was, in fact, yielding phase information which compared favorably with the Tracor series 599R receiver used for control-group data collection purposes, the circuit model was wholly deterministic.

The generation of zero-crossing information for the Omega signal in noise, however, was necessarily stochastic in nature. A complete description of the signal-generating procedures is given in part VI.

CORDET is based on the technique of periodic scanning; that is, each portion of the DPLL is observed and updated with each clock pulse, significant information about the state of the system is recorded, and the process is repeated. In the case of CORDET, the significant information includes the states of all logic circuit elements, including phase difference information referenced to a local oscillator. The phase information is, of course, of primary importance in an Omega navigation receiver. The ability to compare the DPLL

output with the true phase of the simulated Omega signal, and to provide graphical output, are advantages of a computer simulation over a laboratory hardware experiment. A user's technical description is provided in the Appendix.

IV DESCRIPTION OF CIRCUIT

The DPLL follows closely the ideas of J.M. Clark[1], first presented in 1968. The DPLL presently in use at Ohio University is essentially a highly adapted version of his digital phase tracking filter. A detailed discussion of the loop circuit operation is given by Chamberlin[2].

The circuit employs a 6-bit counter which cycles once every 64 pulses of the 652.8 KHz clock (see Figure 1). This counter therefore cycles at a rate of once each 10.2 KHz (=652.8 KHz ÷ 64). The number in the counter is compared continuously to the six most significant bits in the up-down counter, which is in turn fed by the phase detector. During the brief period when the numbers in the two counters are bit-wise identical, the output of the comparator goes to the logical 1 state. Although the duration of this pulse may vary, the uni-directional counter generally advances at a much faster rate, indicating a pulse width of 1/652.8 KHz, or approximately 15 μsec. This pulse is used to trigger a monostable, creating a window whenever the contents of the two registers are identical.

When viewed through this window, the zero crossings of the incoming signal are seen to be either leading or lagging in relative phase. When a lag is indicated, the up-down counter receives a down-count command, and the next window is created earlier in time. In this way the edge of the window gating function is brought more and more closely into coincidence with the zero crossings of the true Omega signal. Similarly, when a phase lead is indicated, count-up commands cause the window to be created later in time.

V THE DPLL AS A CROSS-CORRELATION DEVICE

In a cross-correlation type receiver scheme, two signals are combined to give a d.c. level by first delaying one signal in time, multiplying their resulting signals, and integrating or lowpass filtering (see Figure 2). To see how the DPLL performs this same function in an all-digital sense, it is first necessary to view the circuit operation under no-noise conditions. Under these conditions there is no phase shift due to noise perturbations, nor is there a Doppler shift. The only phase error is then due to initial phase difference at turn-on.

Assume that the initial phase difference is such that a count-up command is given to the up-down counter. If N bits are being integrated, there will be a delay of 2^n up-count commands before the comparator logic "sees" a difference in time between successive occurrences of identical numbers in the two counters. The time delay in such a case is $(2^n/10.2 \times 10^3)$ seconds.

The output of the comparator, θ, is the estimate of the Omega signal phase, θ_Ω. When lock is obtained, $\theta = \theta_\Omega$, and the binary number contained in the six most significant bits of the up-down counter is the phase difference, analogous to the d.c. level obtained in an analog cross-correlation device.

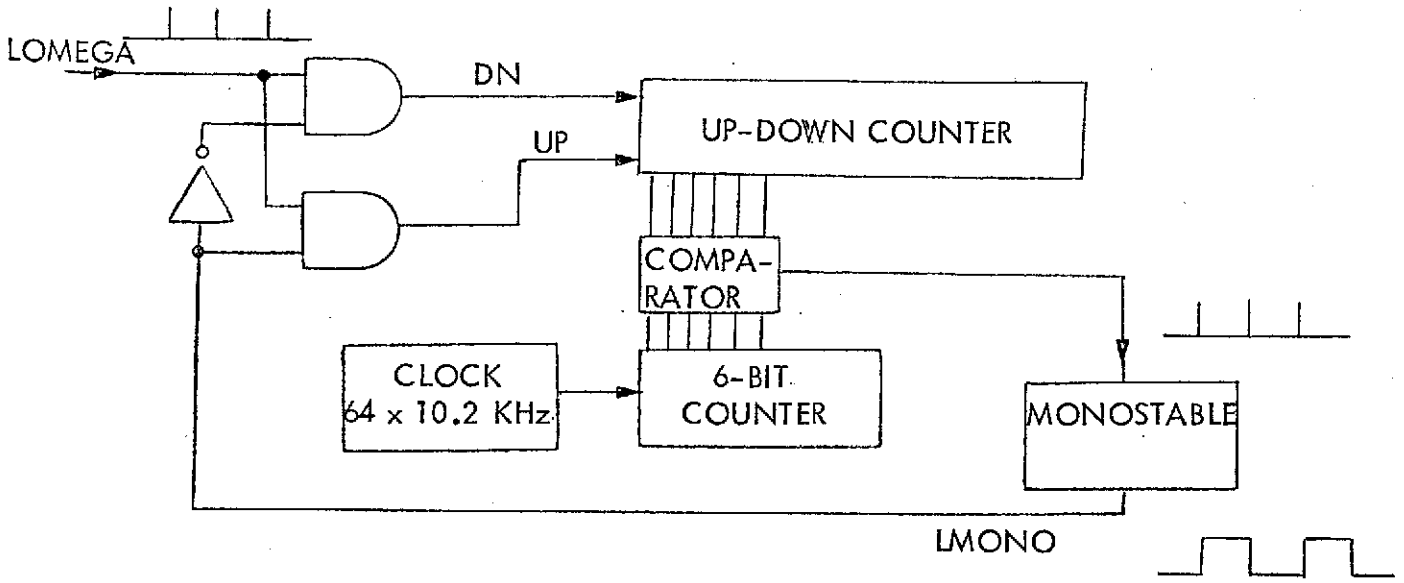


Figure 1. The DPLL Circuit.

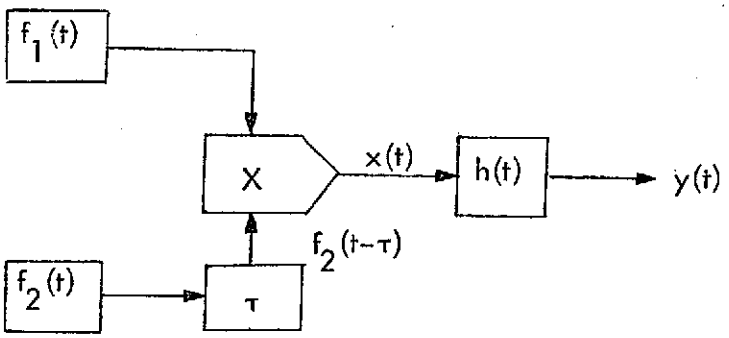


Figure 2. A Generalized Cross-Correlation Device.

VI SIGNAL INPUT SIMULATION

Since the receiver front end provides a 15 Hz bandwidth narraw-band filter, the derived phase information can be considered to result from a noise-plus-signal input to a narrow-band system.

The composite input signal can be expressed as

$$g(t) = f(t) + A\cos(\omega_m t + \psi) \tag{1}$$

where ω_m is the center frequency of the filter, and $f(t)$ is the narrow-band noise process

$$f(t) = E(t)\cos(\omega_m t + \phi(t)). \tag{2}$$

It can be shown [3] that the amplitude process $E(t)$ is Rayleigh distributed, while the phase process $\phi(t)$ is uniformly distributed on the interval $(0, 2\pi)$.

Now, $f(t)$ can be written in the form defining its quadrature components,

$$f(t) = e_c(t)\cos\omega_m t + e_s(t)\sin\omega_m t \tag{3}$$

where $e_c(t)$ and $e_s(t)$ are independent Gaussian processes [3].

From equation 1, $g(t)$ may be expressed as

$$g(t) = f(t) + A\cos\psi\cos\omega_m t + A\sin\psi\sin\omega_m t. \tag{4}$$

Substituting for $f(t)$, and using

$$\cos(a + b) = \cos a\cos b - \sin a\sin b,$$

we have

$$g(t) = (e_c(t) + A\cos\psi)\cos\omega_m t + (e_s(t) + A\sin\psi)\sin\omega_m t. \tag{5}$$

Equation 5 may be put in the form

$$g(t) = R(t)\cos(\omega_m t + \vartheta(t)) \tag{6}$$

where $$R(t) = \{[e_c(t) + A\cos\psi]^2 + [e_s(t) + A\sin\psi]^2\}^{\frac{1}{2}} \tag{7}$$

and

$$\theta(t) = \arctan \frac{e_s(t) + A \sin \psi}{e_c(t) + A \cos \psi} \qquad (8)$$

Equation 8 is therefore the input phase representation to the phase detector, where $e_c(t)$ and $e_s(t)$ are statistically-independent Gaussian white noise processes with zero mean and unit variance, and A is the normalized signal amplitude. In general, ψ can be a function of time to represent the Doppler shift in the dynamic case. At speeds less than Mach I, however, the Doppler shift is well below the 5.06° (1/64 of a 10.2 KHz cycle) quantizing interval of the DPLL, and so ψ can be considered constant over the sampling interval.

A block diagram of the noise input phase simulation is given in Figure 3.

VII CONCLUSIONS

In determining the optimal integration time of the first order loop, consider the case of a small aircraft in a noisy environment. As the time-multiplexed Omega transmission sequence progresses, the aircraft will change position between successive transmitting time intervals of the 10.2 KHz frequency. For an aircraft flying a course coincident with a station pair baseline (worst case), the change in position will amount to 12° of 10.2 KHz phase at a speed of 200 mph, conservative for private aircraft. For this reason, step phase inputs of 12° were taken both in a noise environment of 10 decibels SNR and in the no-noise case.

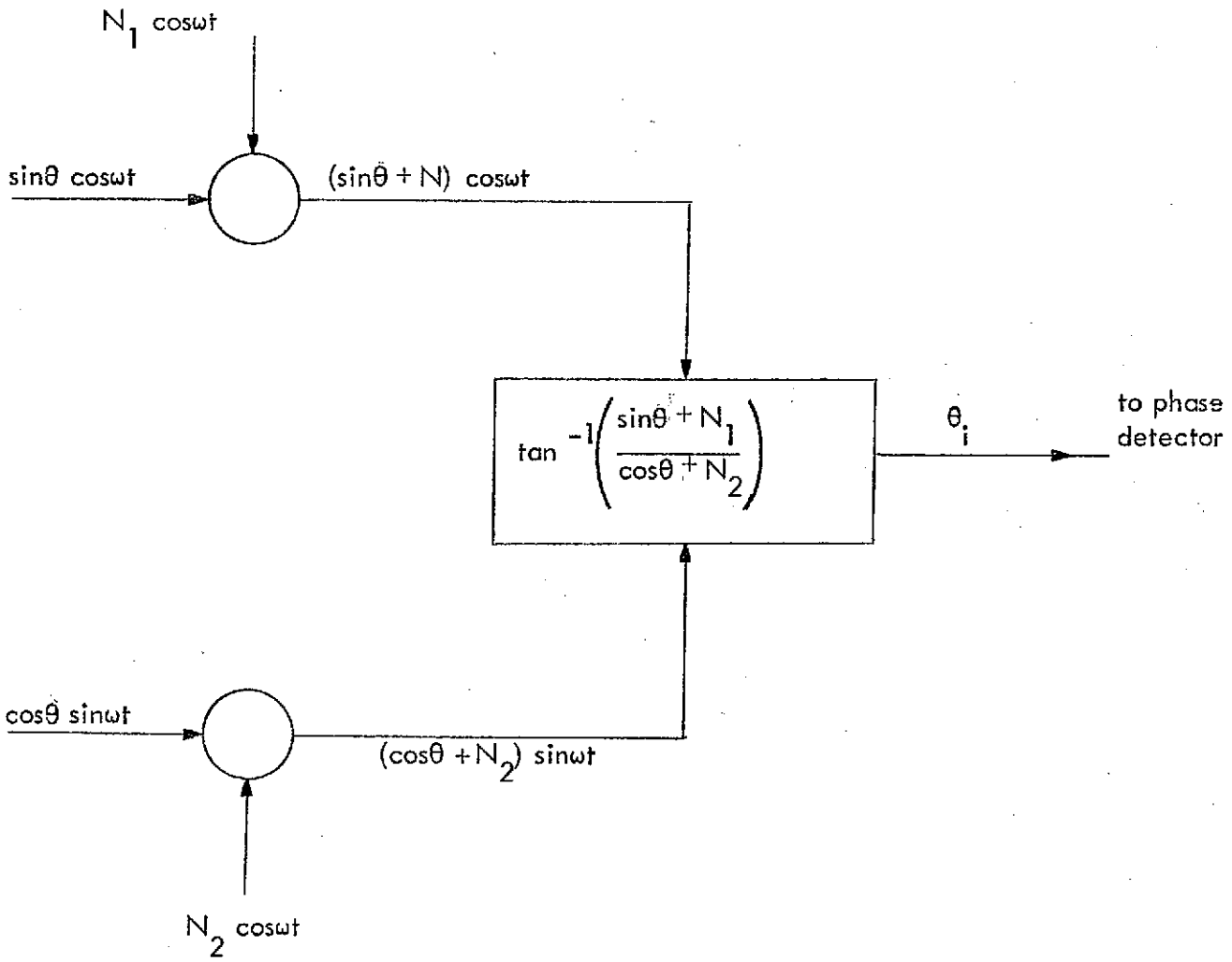
To determine optimal integration time, phase error at the end of the 625 msec. sampling interval is plotted vs. integration time in Figures 4 and 5. As could be expected, the best integration time in noise is one which is as long as possible without leaving a residual phase error at the end of a sampling interval in the no-noise case. For aircraft at 200 mph, 0.1 sec., or 10 bits of integration proves to be optimal.

VIII SUMMARY

The Fortran program CORDET can prove to be a useful tool in analysis and design of a first order all-digital phase-lock loop. The DPLL simulated in CORDET is suggested as useful for designers of digital processing Omega receivers or other VLF signal processing systems.

IX ACKNOWLEDGEMENTS

Research on Omega at Ohio University is funded by NASA. This work is being done at the Avionics Engineering Center headed by Dr. Richard H. McFarland. Mr. Ralph Burhans is project engineer and Dr. R. W. Lilley is staff consultant on the Omega project. The author wishes to thank these persons for their assistance in the preparation of this paper. Additional assistance was provided by K. A. Chamberlin and J. E. Grover, graduate



N_1 and N_2 are statistically-independent Gaussian white noise processes with zero means and unit variances.

Figure 3. Block Diagram Simulation of Narrow-Band Noise.

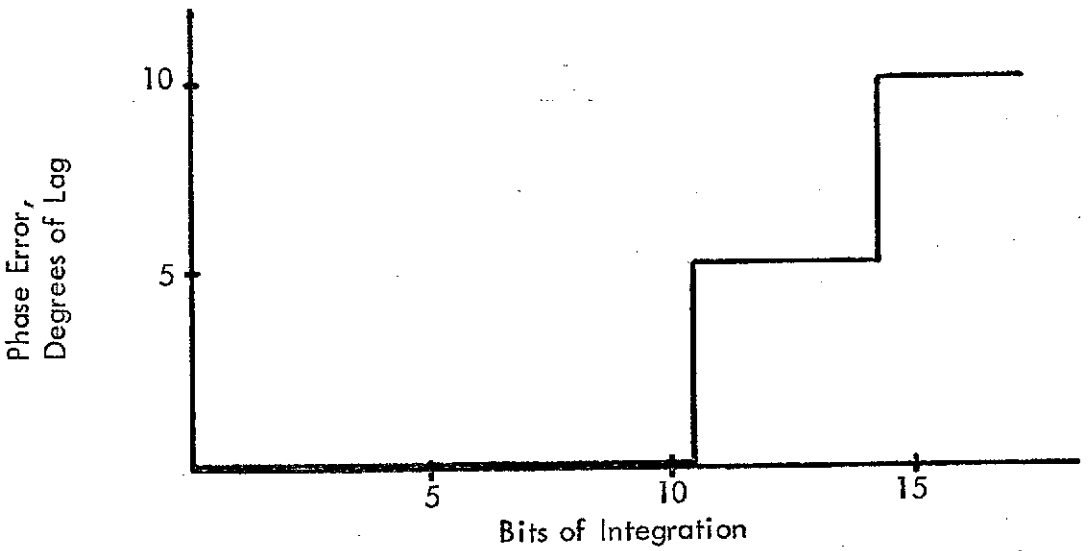


Figure 4. Phase Error at End of Sampling Interval vs. Bits of Integration, No-Noise Case.

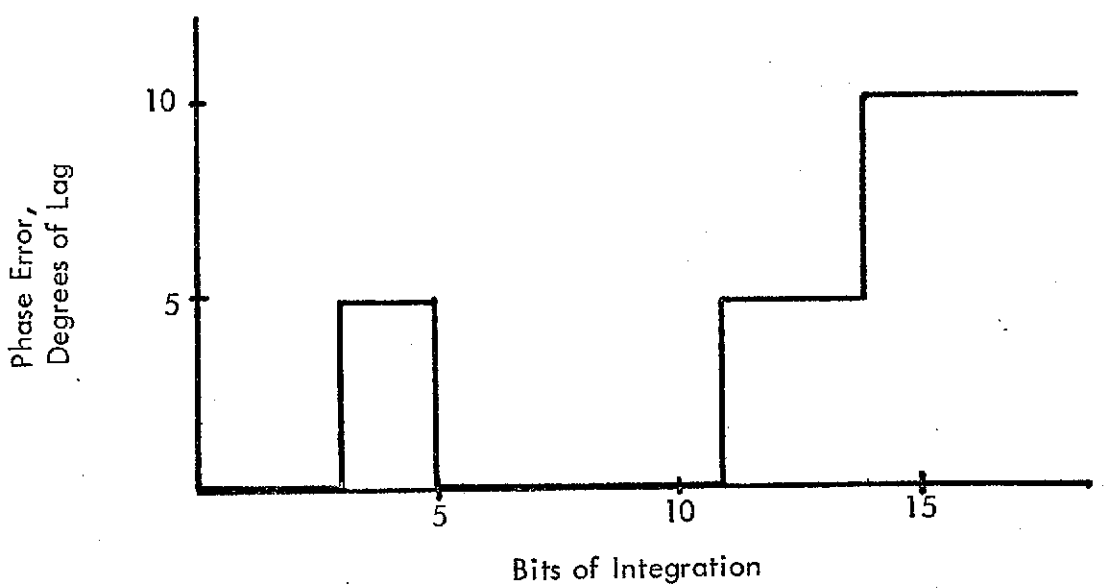


Figure 5. Typical Plot of Phase Error at End of Sampling Interval vs. Bits of Integration, 10 Decibel SNR.

assistants, D. G. Herold, staff engineer, and Dr. J. E. Essman, Assistant Chairman, Department of Electrical Engineering at Ohio University.

X REFERENCES

[1] Clark, John M., "Aircraft Navigation Using Omega", IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-5, No. 5, September, 1969.

[2] Chamberlin, Kent A., "The Memory-Aided Phase-Locked Loop," paper presented at Second Omega Symposium, ION, Washington, D. C., November 7, 1974.

[3] Thomas, John B., Statistical Communication Theory, New York, John Wiley + Sons, Inc., 1969.

[4] Weinberg, A. and B. Liu, "Discrete Time Analyses of Nonuniform Sampling First- and Second-Order Digital Phase Lock Loops," IEEE Transactions on Communications, Vol. COM-27, No. 2, February, 1974.

[5] Garrett, P. H., "Optimum Adaptive Phase Estimation Receiver for One-Way Ranging Aircraft Navigation," Technical Report ECOM-0084-S, United States Army Electronics Command, Fort Monmouth, N. J., October, 1972.

[6] Garodnick, J., et al., "Response of an All-Digital Phase Locked Loop," IEEE Transactions on Communications, Vol. COM-22, No. 6, June, 1974.

XI APPENDIX: USER'S TECHNICAL DESCRIPTION OF FORTRAN PROGRAM CORDET

A. General Description.

CORDET is a logic simulation of the all-digital phase-lock loop (DPLL) presently employed in the Omega receiver prototype.

Output provides graphic display of the timing diagrams of the circuit, plots of phase difference vs. time, and a histogram of phase difference between the DPLL output and the object-lock (input) frequency.

Provisions have been made to vary the number of bits of integration, the signal-to-noise ratio of the input signal, and the initial phase difference between the DPLL output and the object-lock frequency.

B. Capabilities and Limitations.

1. The program assumes the presence of a local oscillator in the circuit with a frequency of 64 times the object-lock frequency.

2. Compile and load time is 15 seconds. Execute time is approximately 0.15 seconds per cycle of object-lock frequency specified (when only phase difference and histogram are plotted).

3. Approximately 21×10^3 bytes of storage are used.

4. Subroutines HISTG and RANDNR are called from the Ohio University computer center request-call library. RANDNR uses the multiplicative congruential method of pseudo-random number generation. HISTG is a plotting subroutine.

C. Technical Description.

Language:	FORTRAN IV
Number of arguments:	11
1. Data card 1:	NCYC, MNUMB, NNUMB, DCYC, DELPHI, NOISE, A
Mode:	INTEGER (1, 2, 3, 6), REAL (4, 5, 7)
2. Data card 2:	TD, PHA, HISTOG
Mode:	INTEGER
3. Data card 3:	TITLE
Mode:	ALPHANUMERIC

Argument definitions:

NCYC	-	is the number of object-frequency cycles to be considered.
MNUMB	-	is the number of bits in the up-counter.
NNUMB	-	is the total number of bits in the up-down counter.
DCYC	-	is the duty cycle of the monostable.
DELPHI	-	is the initial phase difference in degrees of lag (DPLL output lags input).
NOISE	-	is zero for no-noise case, 1 otherwise.
A	-	is voltage signal-to-noise ratio in rational number form (not db).
TD	-	is 1 if timing diagrams are desired, 0 otherwise.
PHA	-	is 1 if phase difference vs. time plots are desired, 0 otherwise.
HISTOG	-	is 1 if histogram is desired, 0 otherwise.
TITLE	-	is title for histogram plot.

Figure A-1 represents the positive-logic implementation of the DPLL. CORDET performs one pass through a DO-loop for each pulse of the 64 x 10.2 KHz clock. New values for the logical variables indicated in the diagram are assigned on each pass.

Changes in the DPLL circuit will of course necessitate changes in the logical statements in CORDET's main program. These changes are facilitated by liberal use of comment cards in the program deck.

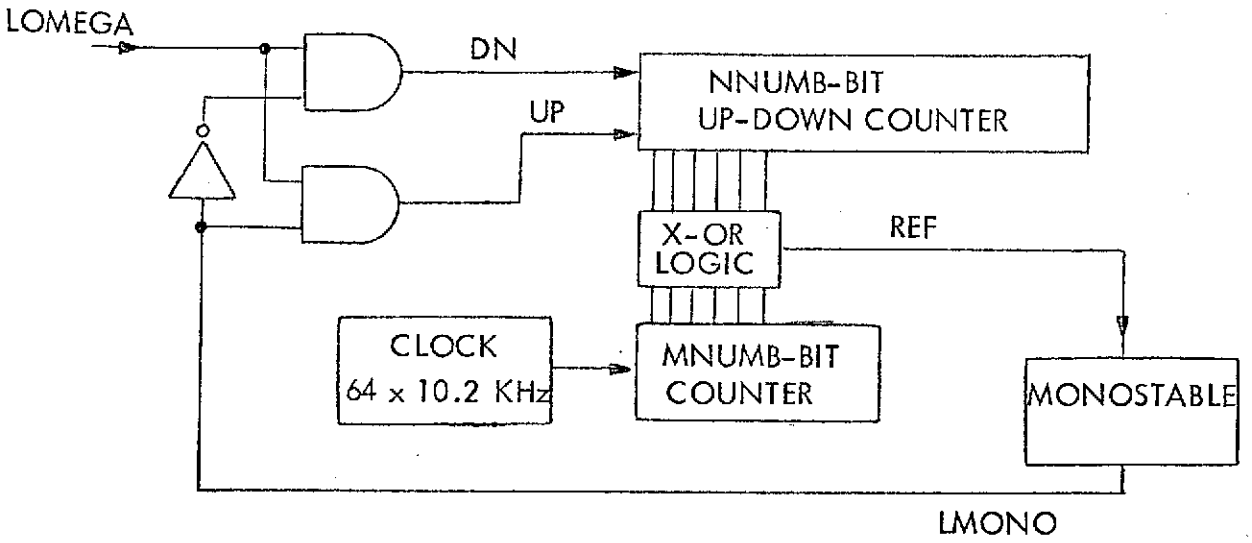


Figure A-1. Positive-Logic Implementation of the DPLL.

D. Program Listing and Sample Outputs.

LABEL MAP

LABEL	LOCATION	LABEL	LOCATION	LABEL	LOCATION	LABEL	LOCATION	LABEL	LOCATION
1	0003AE	2	0003F4	3	000418	4	000434	5	000450
6	00046C	7	000488	8	0004A4	9	0004C0	10	0004DC
11	0004F8	12	00053C						

TOTAL MEMORY REQUIREMENTS 000594 BYTES PLOT1
/DATA

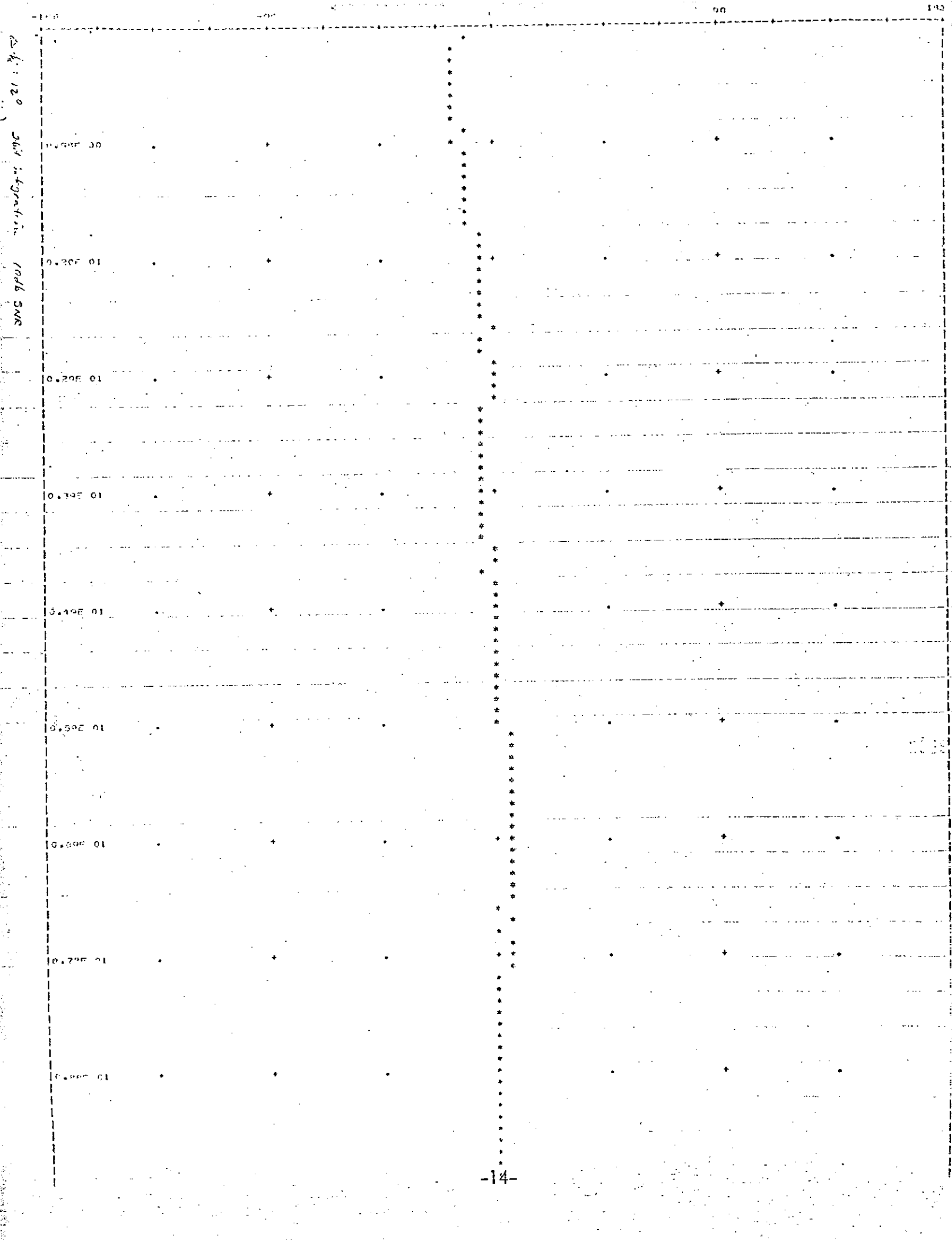
MODULE	ADDR	LENGTH	ENTRY POINTS (E), EXTERNAL (X) AND COMMON (C) REFERENCES					
MAIN##*	0F000		E-MAIN##	0F000	X-IBCOM#*	X-TRUN	X-MONOS	
		00604	X-OMEGA1		X-PLOT1	X-FIXPI#*		
TRUN#*	0F608	003B8	E-TRUN	0F608	X-FIXPI#*			
OMEGA1#*	0F9C0	001E4	E-OMEGA1	0F9C0				
MONOS#*	0FBA8	00274	E-MONOS	0FBA8				
PLOT1#*	0FE20	00594	E-PLOT1	0FE20	X-IBCOM#*			

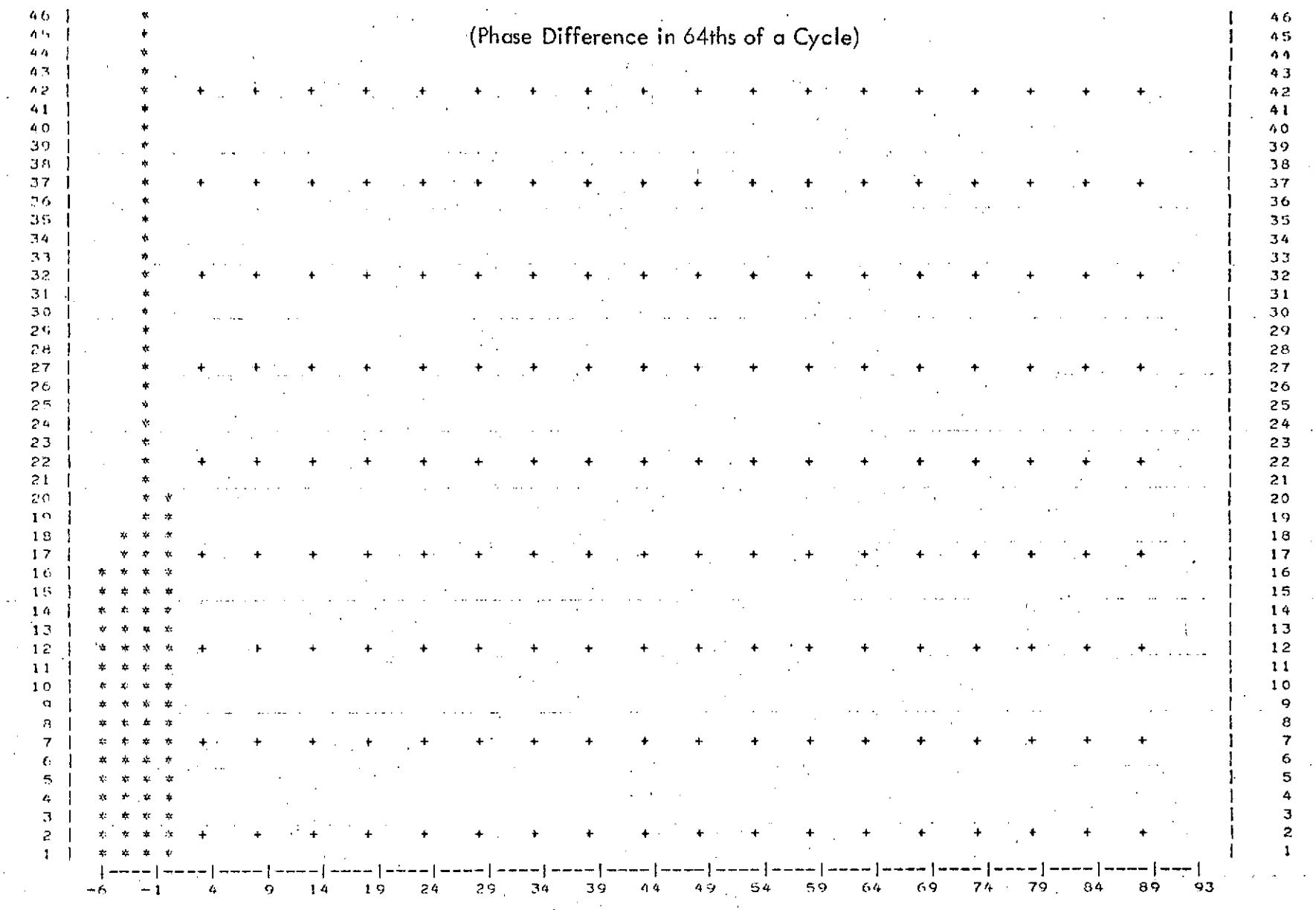
MODULES LOADED FROM AUTO - CALL LIBRARY

BOAFIXPI 10388 000A8 X-IBCOM# E-FIXPI# 103C0
MODULE SUCCESSFULLY LOADED --- 2FB58 BYTES OF STORAGE REMAINING / EXECUTION BEGUN AT 0F000

40 3 5 0.50 90.00 0.13

I	N	NTRUN	M	REF	LOMEGA	POMEGA	UP	ON	LMONO	LMONO2	INDLOC	EX
1	0	0	1	|	|	|	|	|	|	|	|	|
2	31	0	2	|	|	|	|	|	|	|	|	|
3	31	7	3	|	|	|	|	|	|	|	|	|
4	31	7	4	|	|	|	|	|	|	|	|	|
5	31	7	5	|	|	|	|	|	|	|	|	|
6	31	7	6	|	|	|	|	|	|	|	|	|
7	31	7	7	|	|	|	|	|	|	|	|	|
8	31	7	0	|	|	|	|	|	|	|	|	|
9	31	7	1	|	|	|	|	|	|	|	|	|
10	32	7	2	|	|	|	|	|	|	|	|	|
11	32	0	3	|	|	|	|	|	|	|	|	|
12	32	0	4	|	|	|	|	|	|	|	|	|
13	32	0	5	|	|	|	|	|	|	|	|	|
14	32	0	6	|	|	|	|	|	|	|	|	|
15	32	0	7	|	|	|	|	|	|	|	|	|
16	32	0	0	|	|	|	|	|	|	|	|	|
17	32	0	1	|	|	|	|	|	|	|	|	|
18	33	0	2	|	|	|	|	|	|	|	|	|
19	33	0	3	|	|	|	|	|	|	|	|	|
20	33	0	4	|	|	|	|	|	|	|	|	|
21	33	0	5	|	|	|	|	|	|	|	|	|
22	33	0	6	|	|	|	|	|	|	|	|	|
23	33	0	7	|	|	|	|	|	|	|	|	|
24	33	0	0	|	|	|	|	|	|	|	|	|
25	33	0	1	|	|	|	|	|	|	|	|	|
26	34	0	2	|	|	|	|	|	|	|	|	|
27	34	0	3	|	|	|	|	|	|	|	|	|
28	34	0	4	|	|	|	|	|	|	|	|	|
29	34	0	5	|	|	|	|	|	|	|	|	|
30	34	0	6	|	|	|	|	|	|	|	|	|
31	34	0	7	|	|	|	|	|	|	|	|	|
32	34	0	0	|	|	|	|	|	|	|	|	|
33	34	0	1	|	|	|	|	|	|	|	|	|
34	35	0	2	|	|	|	|	|	|	|	|	|
35	35	0	3	|	|	|	|	|	|	|	|	|
36	35	0	4	|	|	|	|	|	|	|	|	|
37	35	0	5	|	|	|	|	|	|	|	|	|
38	35	0	6	|	|	|	|	|	|	|	|	|
39	35	0	7	|	|	|	|	|	|	|	|	|
40	35	0	0	|	|	|	|	|	|	|	|	|
41	35	0	1	|	|	|	|	|	|	|	|	|
42	36	0	2	|	|	|	|	|	|	|	|	|
43	36	1	3	|	|	|	|	|	|	|	|	|
44	36	1	4	|	|	|	|	|	|	|	|	|
45	36	1	5	|	|	|	|	|	|	|	|	|
46	36	1	6	|	|	|	|	|	|	|	|	|
47	36	1	7	|	|	|	|	|	|	|	|	|
48	36	1	0	|	|	|	|	|	|	|	|	|
49	36	1	1	|	|	|	|	|	|	|	|	|
50	37	1	2	|	|	|	|	|	|	|	|	|
51	37	1	3	|	|	|	|	|	|	|	|	|
52	37	1	4	|	|	|	|	|	|	|	|	|
53	37	1	5	|	|	|	|	|	|	|	|	|
54	37	1	6	|	|	|	|	|	|	|	|	|
55	37	1	7	|	|	|	|	|	|	|	|	|
56	37	1	0	|	|	|	|	|	|	|	|	|
57	37	1	1	|	|	|	|	|	|	|	|	|
58	38	1	2	|	|	|	|	|	|	|	|	|
59	38	1	3	|	|	|	|	|	|	|	|	|
60	38	1	4	|	|	|	|	|	|	|	|	|
61	38	1	5	|	|	|	|	|	|	|	|	|


0.30E 00
0.30E 01
0.30E 01
0.30E 01
0.40E 01
0.50E 01
0.60E 01
0.70E 01



NUMBER OF POINTS REPRESENTED IS 100 NUMBER OF POINTS TO REPRESENT ONE * = 1

ACTUAL PLOTTED DATA LIMITS ARE FROM -6 TO 0 INTERVAL FOR THE GRAPH IS 1

VAL.	FREQ	VAL.	FREQ	VAL.	FREQ	VAL.	FREQ	VAL.	FREQ	VAL.	FREQ	VAL.	FREQ	VAL.	FREQ	VAL.	FREQ
-6	16	-4	18	-2	46	0	20										

```
C----------INITIATE AND DIMENSION---------
      INTEGER*2  REF,S(24),INDI,NS,TD,PHA,HISTOG
      INTEGER PHADIF,PH(650)
      LOGICAL LMONO,UP,INDLOC,UPPE,DN,LOMEGA,IND,LMONO2,EX,POMEGA
C----------OM AND AM MUST BE DIMENSIONED TO 2*(2**MNUMB)+1----------
      LOGICAL*1 OM(129),AM(129)
      DIMENSION ARRAY(650)
      EX = .FALSE.
      LMONO2 = .FALSE.
      INDLOC = .FALSE.
      POMEGA = .FALSE.
      READ (1,13) NCYC,MNUMB,NNUMB,DCYC,DELPHI,NOISE,A
13    FORMAT (3I5,2F10.2,I2,F5.2)
      READ (1,14) TD,PHA,HISTOG
14    FORMAT (3I5)
C----------THETA IS INITIAL PHASE SHIFT,=2*PI*DELPHI/360.-------
      THETA = DELPHI*0.01745329
      SINE = SIN(THETA)
      COSINE = COS (THETA)
      DO 57 I=1,129
      OM(I) =.FALSE.
      AM(I)=.FALSE.
57    CONTINUE
      JJ = 0
      JJJ = 0
      II = 0
      MONO = 0
      M = 0
      N = 0
      MREGM = 2**MNUMB
      MREGN = 2**NNUMB
      HALF = FLOAT(MREGM)/2.
      MHALF = HALF
      K = MREGM
      NCYCP = NCYC*MREGM
      NCYCT = NCYC*.1
      X = (DELPHI/360.)*MREGM
      IX = X
      III = 0
      JII = 64
      INDCH = 0
C----------DO-LOOP BEGINS WITH OCCURRENCE OF FIRST CLOCK PULSE------
      DO 12 I=1,NCYCP
C---------M IS NUMBER STORED IN UP-COUNTER---------
      M = M+1
      IF (M-MREGM) 10,11,11
11    M = 0
10    CONTINUE
C--------BINARY NUMBER IN UP-DN COUNTER IS TRUNCATED DUE TO BIT
C----------INTEGRATION.   NTRUN IS RETURNED----------
      CALL TRUN (NNUMB,MNUMB,N,NTRUN,B)
C--------IF M=NTRUN, REF GOES HIGH
      IF (M-NTRUN) 1,2,1
1     REF = 0
      GO TO 3
2     REF = 1
3     CONTINUE
C----------MONOS PROVIDES OUTPUT OF MONOSTABLE.  K IS TIMER FOR MONO.
C----------LMONO IS LOGICAL OUTPUT OF MONOSTABLE.
      K = K+1
      CALL MONOS (K,REF,DCYC,MREGM,LMONO)
C--------LOMEGA IS OMEGA SIGNAL ZERO CROSSINGS.
C----------A IS VOLTAGE SNR.  POMEGA IS TRUE PHASE OF OMEGA SIGNAL.
      IF (IX.EQ.M) GO TO 30
      POMEGA = .FALSE.
      GO TO 31
30    POMEGA = .TRUE.
31    CONTINUE
      IF (NOISE.EQ.0) GO TO 32
      IF (POMEGA) GO TO 34
      GO TO 51
34    INDCH = -1*INDCH+1
      IF (INDCH.EQ.1) GO TO 37
      DO 38 IJK = 1,129
      AM(IJK) =.FALSE.
38    CONTINUE
C--------NOISE IS ADDED TO QUADRATURE COMPONENTS OF OMEGA SIGNAL.-----
39    CALL ANORM (AN)
      ANUM = A*SINE+AN
      CALL ANORM (AN)
```

```
      DENOM = A*COSINE+AN
      THETAI = ATAN2 (ANUM,DENOM)
      PCYC = (THETAI/6.28318)*MREGM
      INDEX = PCYC
      IF (INDCH.EQ.1) GO TO 50
      AM (MREGM+INDEX) =.TRUE.
      GO TO 51
50    OM (MREGM+INDEX) =.TRUE.
      GO TO 51
37    DO 52 IJK=1,129
      OM(IJK) =.FALSE.
52    CONTINUE
      GO TO 39
51    CONTINUE
      JII = JII+1
      III = III+1
      IF (III.EQ.129) III = 1
      IF(JII.EQ.129) JII =1
      LOMEGA = OM(III).OR.AM(JII)
      IF ((I.EQ.M).AND.(POMEGA)) LOMEGA = POMEGA
32    IF (NOISE.EQ.0) LOMEGA = POMEGA
C-----LMONO AND LOMEGA ENTER PHASE DETECTOR-----------
      UP = LMONO.AND.LOMEGA
      DN =(.NOT. LMONO).AND.LOMEGA
C---------PHASE DETECTOR OUTPUT FEEDS UP-DN COUNTER
      IF (UP) N = N+1
      IF (DN) N = N-1
C----------COUNT REGISTER MUST CONTAIN POSITIVE NUMBER
      IF(N) 8,9,9
8     N = MREGN+N
9     CONTINUE
C----------UP-DN COUNTER COUNTS MODULO-MREGN----------
      IF (MREGN-N) 23,24,24
23    N=0
24    CONTINUE
      IF (TD.EQ.0) GO TO 501
      IF (I-1) 40,40,70
40    CONTINUE
C---------TIMING DIAGRAM IS PLOTTED----------
      PRINT 500
500   FORMAT('1',2X,'I',3X,'N',3X,'NTRUN',3X,'M',4X,'REF',3X,'LOMEGA',2X
     *,'POMEGA',4X,'UP',6X,'DN',4X,'LMONO',3X,'LMONO2',2X,'INDLOC',4X,'E
     *X'//)
70    CONTINUE
      CALL PLOT1 (I,N,NTRUN,M,REF,LOMEGA,LMONO,LMONO2,DN,UP,INDLOC,EX,PO
     *MEGA)
501   CONTINUE
      IF ((PHA.EQ.0).AND.(HISTOG.EQ.0)) GO TO 12
      IF (.NOT.POMEGA) GO TO 65
      JJ = JJ+1
      IF (JJ.EQ.1) GO TO 28
      AJ = JJ/10.
      IJ = AJ
      IF (IJ.NE.AJ) GO TO 65
28    CONTINUE
      IF (M.LT.MHALF) GO TO 20
      IF (NTRUN.LT.MHALF) GO TO 21
22    PHADIF = 2*(NTRUN-M)
      GO TO 45
21    PHADIF = (NTRUN + MREGM-M)*2
      GO TO 45
20    IF (NTRUN.LT.MHALF) GO TO 22
      PHADIF = 2*(NTRUN-MREGM-M)
45    CONTINUE
C-----------ARRAY IS LOADED FOR PLOT SUBROUTINE----------
      JJJ = JJJ+1
      PH(JJJ) = PHADIF
65    CONTINUE
12    CONTINUE
      IF (PHA.EQ.0) GO TO 502
      CALL PLOT (PH,JJJ)
502   CONTINUE
      IF (HISTOG.EQ.0) GO TO 503
C----------ARRAY IS LOADED FOR HISTG SUBROUTINE
      DO 27 I = 1,JJJ
      ARRAY (I) =  FLOAT(PH(I))
27    CONTINUE
      DIMENSION TITLE(20)
      READ (1,146) TITLE
146   FORMAT (20A4)
```

```
      DATA AST/'*'/
C-----------HISTOGRAM IS PLOTTED-----------
      CALL HISTG   (JJJ,ARRAY,TITLE,1,AST)
503   CONTINUE
      STOP
      END

      SUBROUTINE TRUN (NNUMB,MNUMB,N,NTRUN,B)
C        THIS SUBROUTINE CONVERTS DECIMAL INTEGERS TO BINARY FORM
C           AND TRUNCATES, SHIFTS DOWN REMAINING BITS AND RETURNS A DECIM
      INTEGER*2 B(NNUMB)
      DO 9 I = 1,NNUMB
      B(I) = 0
9     CONTINUE
      I=0
      DUM = N
1     DUM=DUM/2.
      I = I+1
      IDUM=DUM
      IF (DUM-IDUM) 2,3,2
2     B(I) = 1
      DUM=DUM- .5
      IF (DUM) 4,4,1
3     B(I) = 0
      IF (DUM) 4,4,1
4     CONTINUE
C     THE BINARY NUMBER IS TRUNCATED
      DO 11 I = 1,MNUMB
      B(I) = B(I+(NNUMB-MNUMB))
11    CONTINUE
C     THE TRUNCATED BINARY NUMBER IS CONVERTED BACK TO DECIMAL
      NTRUN = 0
      DO 10 I=1,MNUMB
      K=B(I)*(2**(I-1))
      NTRUN = NTRUN + K
10    CONTINUE
      RETURN
      END


      SUBROUTINE MONOS (K,REF,DCYC,MREGM,LMONO)
      LOGICAL LMONO
      INTEGER*2 REF
      X = DCYC*MREGM - 1.
      IX = X
      IF (REF) 10,10,12
10    IF (K-IX) 14,14,11
11    MONO = 0
      GO TO 18
12    IF (K-IX) 14,14,19
19    MONO = 1
      K=0
      GO TO 18
14    MONO = 1
18    IF (MONO)15,15,16
15    LMONO = .FALSE.
      GO TO 17
16    LMONO = .TRUE.
17    RETURN
      END
```

```
      SUBROUTINE PLOT (PH,JJJ)
      REAL LINE(129)
      INTEGER PHADIF,PH(JJJ)
      DATA DOT/'.'/,PLUS/'+'/,BLK/' '/,VLINE/'|'/,AST/'*'/
      DO 12 JJ = 1,JJJ
      IF (JJ.NE.1) GO TO 5
      PRINT 8
    8 FORMAT  ('1',45X,'PHASE DIFFERENCE IN DEGREES VS. TIME IN MSEC.')
    4 PRINT 2
    2 FORMAT   (3X,'-180',28X,'-90',30X,'0',31X,'90',28X,'180')
    6 PRINT 7
    7 FORMAT (4X,'+',16('-------+'))
    5 CONTINUE
      NSP = PH(JJ) + 65
      IF (NSP.GT.129) GO TO 15
      IF (NSP.LT.1) GO TO 15
      GO TO 3
   15 CONTINUE
      PRINT 23,PH(JJ)
   23 FORMAT (100X,'PH= ',I10)
      RETURN
    3 CONTINUE
C        CLEAR LINE
```

```
      DO 1 J = 1,129
      LINE(J) = BLK
    1 CONTINUE
      LINE(1) = VLINE
      LINE(129) = VLINE
      AI = JJ/10.
      II = AI
      IF(AI.NE.II) GO TO 9
      LINE(17) = DOT
      LINE(33) = PLUS
      LINE(49) = DOT
      LINE (65) = PLUS
      LINE (81) = DOT
      LINE (97) = PLUS
      LINE (113) = DOT
    9 CONTINUE
   10 LINE(NSP) = AST
      IF (AI.EQ.II) GO TO 20
      WRITE(3,11) LINE
   11 FORMAT (4X,129A1)
      GO TO 12
   20 TIME =(FLOAT(JJ)*.0098039216)
      WRITE (3,21) LINE
   21 FORMAT (4X,129A1)
      WRITE (3,22) TIME
   22 FORMAT ('+',4X,E8.2)
   12 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE PLOT1(I,N,NTRUN,M,REF,LOMEGA,LMONO,LMONO2,DN,UP,INDLOC,
     *EX,POMEGA)
      INTEGER*2 REF
      LOGICAL LMONO,UP,INDLOC,DN,LOMEGA,LMONO2,EX,POMEGA
      INTEGER LINE(66)
      INTEGER BLK/' '/,VLINE/'|'/
      DO 1 J = 1,66
      LINE(J) = BLK
    1 CONTINUE
      LINE(1) = VLINE
      DO 2 K = 1,8
      LINE (8*K) = VLINE
    2 CONTINUE
      IF (REF.NE.1) GO TO 3
      LINE(3) = VLINE
      LINE (1) = BLK
    3 IF (.NOT.LOMEGA) GO TO 4
      LINE (10) = VLINE
      LINE (8) = BLK
    4 IF (.NOT.POMEGA) GO TO 5
      LINE(18)= VLINE
      LINE (16)= BLK
    5 IF(.NOT.UP) GO TO 6
      LINE(26)= VLINE
      LINE (24)= BLK
    6 IF(.NOT.DN) GO TO 7
      LINE(34)= VLINE
      LINE (32)= BLK
    7 IF (.NOT.LMONO) GO TO 8
      LINE(42)= VLINE
      LINE (40)= BLK
    8 IF (.NOT.LMONO2) GO TO 9
      LINE(50)= VLINE
      LINE (48)= BLK
    9 IF (.NOT.INDLOC) GO TO 10
      LINE(58)= VLINE
      LINE (56)= BLK
   10 IF (.NOT.EX) GO TO 11
      LINE(66)= VLINE
      LINE (64)= BLK
   11 WRITE(3,12) I,N,NTRUN,M,LINE
   12 FORMAT (1X,2I4,I6,I5,5X,66A1)
      RETURN
      END
  100    6    8      0.50          0.0 1   3.60
    0    1    1
PHASE DIFFERENCE IN 64THS OF A CYCLE
```