NGL-03-002-024

UNIVERSITY OF ARIZONA

COLLEGE OF ENGINEERING

Department of Electrical Engineering

Computer Science Research Laboratory

CSRL Memo No. 197

A Hybrid-computer Data-channel Interface and its Application to the Simulation of Variable Time Delays

by M. Naka

National Aerospace Laboratory, Tokyo

January, 1970

N70-27381	
(ACCESSION NUMBER)	(THRU)
39	Ø
(PAGES)	(CODE)
CR-109831	08
(NASA CR OR TMX OR AD NUMBER)	(CATEGORY)

41p

ABSTRACT

This report describes implementation of a data-channel interface for the ASTRAC II high-speed iterative differential analyzer and the PDP-9 digital computer, and its application to the simulation of variable time delay (e.g. pipe-line delays in aerospace and chemical problems.) The cycle-stealing data-channel interface effects automatic block transfers of data from and to the digital computer at rates of up to 250,000 words per second in response to request pulses from the analog computer. Variable time delays can be implemented if analog data read in and stored at a constant sample rate are read out into a digital-to-analog converter at a variable rate established by a voltage-controlled oscillator patched on the analog computer. A simple integral equation was solved to test implementation of (constant) time delays.

Data Channel Interface

1. General description

The PDP-9 data channel interface for the ASTRAC II analog computer[7] provides a fast data transfer between PDP-9 and ASTRAC II. Requests for data channel transfers from or to the ASTRAC II are honored by the channel at the completion of the PDP-9 instruction in progress.

The channel is controlled by word count and current address registers held in the digital-computer core memory. Both registers are initialized by a program instruction depositing suitable words. When the processor honors a data-channel request, the respective device transmits the address of its assigned word-count register. During the first cycle of the channel transfer, the contents of this word counter are incremented, and the address of the current-address register is established.[3]

In the second cycle, the contents of the current-address register are incremented to establish the effective address of the memory location delivering or receiving the data word.

During the third cycle (or fourth cycle in the case of out-transfer), the actual data transfer occurs. Because of these counting operations, such memory implemented data channels steal three or four computer cycles per data transfer, But still permit maximum word transfer rates exceeding 250,000 words per second at very low cost.

2. Interface Design

The interface shown in Fig. 1.1 can be used to either read data from the PDP-9 into an analog computer or vice versa.

The primary data transfer divice already installed are the AD converter (ADC) with four multiplexed analog inputs (AMMEX), and four multiplying DA converters (MDAC). A Digital Equipment Corporation interface card (W 104) is used to establish data-channel requests.

The circuit operates as follows:

1) Device data request flag.

The device data-channel request flag places a ground level on the flag input when a data-channel transfer is required.

This ground level enables the REQ flip-flop in the W104.

The pulse IO SYNC which occurs at 1 µsec intervals and is available on the IO BUS sets the REQ flip-flop in the W104. This pulls the PDP-9 request line to ground thus, requesting a data-channel break cycle in the computer.

If repetitive data transfers are not desired (or not desired at the maximum possible rate), the device flag must be cleared to prevent false data channel cycles and associated loss of data. In our circuit, the clear flag signal from the W104 is used to reset the device flag.

2) Grant signal.

When the PDP-9 is ready to initiate the requested data-channel break, it issues a grant signal to the device. The trailing edge of this signal produces CLR flag which resets the device flag beforementioned, and also sets the ENA flip-flop gating onto the IO ADDR lines. This flip-flop then enables the ENB flip-flop, which is set with the next IO SYNC pulse. This causes SELECT of the W104 to go to ground. Connected to the force-select input of a W103 device selector, this simulates the W103's proper device code, permitting IOT pulses to pass through it.

Meanwhile, ENB is used to gate a Read request (RD RQ) or write request (WR RQ) level to the PDP-9. In the case of a read request, the transfer pulse IOP2 is issued and the data buffer gated on the bus. In the case of a write request, the IOP4 is issued on the reset cycle.

3) Enable signal

ENA IN of the W104 must be connected to the data-channel enable line on the IO bus running toward the PDP-9. ENA OUT must be connected to the next lower priority device (if any) on the data-channel bus. This arrangement sets up priorities for data-channel requests from two more devices.

Data channel timing diagrams are shown in Fig. 1.2.

4) Linkage patchbay

On the linkage patchbay of the ASTRAC II which the user can patch to meet the requirement of his particular application, we have provided the following terminals (See Fig. 1.3).

REQ; The positive-going request signal from device to the data channel indicates that device requires data channel transfer.

RDENA; Enabling RD RQ signal indicates that the device wishes the transfer to be in the "input" direction.

WR ENA; Enabling WR RQ signal indicating that the device wishes the transfer to be in the "output" direction.

RD ENA and WR ENA; Enabling the add-to-memory operation which is a combination of reading and writing.

INC ENA: Enabling the memory increment operation, i.e., the word specified on the IO ADDR lines is incremented.

IO OFLO: A pulse originating in the channel logic which indicates that the transfer of specified number of data words has been completed.

MEM OFLO: Indicating data-channel transfer complete.

ADDR: 14 gate inputs specifies WC ADDR for RD and WR operation. (ground is "0", open c.c.t. is "1").

DATA IN: 18 Bus-gate inputs into PDP-9 IO bus strobed by IOT2 pulse. (ground is "0", open c.c.t. is "1").

DATA AVAILABLE: IOT4 pulse indicates that requested data is on the IO Bus lines. Usually wired to DAØLOAD or DA1LOAD depending on MDACØ or MDAC1 is currently used.

DAØ LOAD: Input pulse into this termination produces two timing pulses 1.5 μsec apart to transfer output data from the accumulator to the buffer register, and from the buffer register to the device register in the MDACØ.

DA1 LOAD: Same as DAØ LOAD except MDAC1.

B: A pulse to transfer output data from accumulator to buffer register.

D: A pulse to transfer output data from buffer register to device register.

ADC: 12 bit ADC outputs (-3V is "1")

3. Operational description

Figure 1.4 shows an example of a data-channel interface applied to a variable-time-delay system (see below).

Flip-flip B202(1) is set by positive-going pulse, which comes from a circuit patched on the analog computer, after the word-count (WC) and current-count (CA) registers have been initialized by the program. This produces a ground-level pulse to set the REQ flip-flop in the W104. Reset-input terminal of the flip-flop B202(1) and CLR REQ is patched from CLR FLAG when repetitive data transfers are not desired, as in the case of variable time delays. The W104 is connected in a standard configuration. It issues a WR RQ pulse when WR ENA is at ground level, and RDENA is at -3V. Also it issues a RD RQ pulse when RDENA is ground level and WRENA is -3V level.

The IO ADDR lines can be connected for any number of different word-count and current-address registers in memory. But the following core-register assignment for this data channel is used, because only one address bit, that is, IO ADDR 14 line is different between READ and WRITE (Fig. 2.2):

Word count 32, Current address 33 <u>for READ</u>,

Word count 22, Current address 23 <u>for WRITE</u>.

The W104 enables a W103 to allow the IOT2 pulse to strobe data from device and the IOT4 pulse to jam transfer data into the device buffer register using both DATA AVAILABLE and DA LOAD. IO power clear is used to reset all flip-flop, when the computer is first turned on.

Figure 1.5 shows the logic-card module layout for the DATA channel interface and, specifically, for variable time delays. This figure indicates the type of logic module found at each location in the interface mounting panels when viewed from the front of ASTRAC II.

Variable-Time-Delay Simulation

1. General Description

Time delay arises from the finite time it takes to propagate an information (signal) for the source to the receiver in a medium. Should the medium in which the signal is propagated change properties as a function of time, the resulting delay could become variable in time.

The magnitude of the delay in propagated signal is dependent on two quantities: the velocity of propagation (V) and the distance between sender and receiver (D). If either of these quantities varies, have a variable-delay problem.

Typical pipeline problems involve a fixed distance with a continuously-variable propagation velocity. Several methods are available to simulate constant time delays, but few are satisfactory for a continuously variable delay. The system described is a relatively new hybrid analog-digital implementation. The basic idea is to sample a data signal at a constant rate, put the samples into core memory, and read them out at a sampling rate proportional to the flow velocity, as determined by a voltage-controlled oscillator (VCO).[6]

2. Variable-Time-Delay Design

The data input signal is converted to 12 bit binary form in an analog to digital converter (ADC) with constant sampling rate. Samples are stored in a block of core memory of the PDP-9, using a data-channel READ operation. An analog voltage proportional to the rate of change of the desired delay velocity in a pipe is applied to an analog-voltage-controlled oscillator whose output-pulses serve as request pulses for data-channel write operations. The pulse repetition rate determines the rate at which data is taken out of memory, and therefore the time the data is in memory, that is, the delay. The starting address of the CA register for data channel read and data channel write is changed according to the initial time delay required.

The VCO is patched on the analog-computer patchbay (Fig. 2.b). The output of the core memory is converted to analog voltage in a MDAC.

A few modifications of the standard data-channel interface were needed to be used as a variable time-delay system:

1) Generation of DCH RQ signal.

2) Mode selection between RD RQ and WR RQ, and also establishment of priority between RD RQ and WR RQ.

3) Control of two WC and CA register with a single data channel.

4) Generation of a pulse proportional to the velocity of delayed signal.

5) Data-channel register initialization for making an "endless loop" memory.

The first three requirements are met by the variable-time delay logic shown in Fig. 2.1 and Fig. 2.2.

DCH RQ signal must occur when data signal sampling clock (read instruction) or VCO output pulse (write instruction) comes.

Since the WRITE request pulses from the fixed-rate clock and the READ request pulses from the VCO can overlap or even coincide, the pulse-separation circuit of Fig.2.3 was designed to give priority to the WRITE request pulse, and also to produce READ and WRITE request pulse waveform 1 μsec long and synchronized with the PDP-9 IO SYNC timing signal (see also the timing diagram of Fig. 2.3-1).

Since the minimum required time interval between data channel read operation and data channel write operation is 7 μsec, we must seperate sampling clock and VCO pulse from each other more than 7 μsec. Four monostable multivibrators (4 MMV's), two AND-gate, and two OR-gates perform this function of separation.

The 7 μsec minimum required time interval between read and write is determined by the following considerations. After a device flag is raised, it takes two processor cycles (i.e. 2 μsec) to receive the grant signal (DCH GR) from the channel of PDP-9. One processor cycle later, the actual data-channel transfer starts. Each output data-channel transfer steals four processor cycles. Therefore, 7 μsec is needed to complete data channel operations (see Fig. 2.5).

The Mode flip-flop selects between READ request (RD RQ) and WRITE request (WR RQ). An AND-gate receiving the output of the mode flip-flop and the enable signal (ENA) from the W104 card is used to select the address of one of two word-count (WC) registers in core memory; the first corresponds to the WC register for READ (assigned core address 32 in octal notation), and the other corresponds to the WC register for WRITE (assigned core address 22 in octal notation). There is only one bit difference between these two addresses of core memory registers (i.e. IO ADDR line 14), so that the switching logic is simple.

The fourth requirement is met by an analog-voltage-controlled oscillator. Figure 2.6 shows the block diagram of this VCO, which is patched on the analog-computer patchbay. The VCO operating characteristic (input voltage versus output frequency of the VCO) is shown in Fig. 2.7. When the input voltage changes from 0.5 volts to 9.0 volts, the output frequency changes 150 HZ to 9200 HZ (linearity is within $1^{o}/o$). Output frequency of the VCO is adjusted to 5000 HZ when 5 volt of input voltage is applied.

The last requirement is met by simple initialization programs, (i.e., neither program interrupt facility (PI) nor multilevel automatic priority interrupt facility (API) was used). Because the maximum speed of the data Channel read or write is determined by the highest pulse repetition rate of the VCO, and is relatively slow (max p.r.r. is 9200 HZ; therefore its period is 110 μsec). An initialization program of this data channel generally consists of two programs; one is a detection routine when WC register has counted up to zero, the other is a service routine that deposits WC, CA register with suitable words. However, if a primary importance is placed on the speed of data channel transfer, then PI or API should be used.

3. Typical performance

The Data-signal-sampling clock rate must be selected according to both the frequency of the data signal and the desired delay time. The period between read-in pulses depends on the sampling clock. The pulse repetition rate from the VCO is $150 + 1065\ (V-0.5)$ HZ where V is the voltage of input signal to VCO. Since V varies from 0.5 volt to 9.0 volt, this pulse rate varies from 150 pps to 9,200 pps, the period between write-out pulses is 110 μsec to 6 msec. If we use, for example, 1000 words of active core memory for this time delay system, the delay obtained is 1000 times the time period (i.e. the delay time is from 110 μsec to 6 sec.). The more core memory capacity is used, the larger delay time is obtained, although the maximum available core memory capacity will depend on the application. In this case, in order to secure 20 samples per cycle of data signal wave over the whole delay range, the highest frequency of the data signal is about 33 HZ; a delay-bandwidth product of 200 will require 1000 words of"core memory".

Errors in time delay arise from three main sources; 1) The error introduced by the VCO, which has a linearity error of 1.0 %. 2) The maximum delay before the data is taken out of memory and reconstructed into an analog voltage by MDAC, 7 μsec for data channel, 20 μsec for MDAC; total 27 μsec. 3) Error in reconverting the memory output to an analog voltage in the MDAC (0.25 %/Half-scale), and converting input data signal to binary form in ADC (0.1%).

The time-delay error for the entire system is the sum of the above errors.

A Constant-time-delay Application

As a demonstration of the use of constant time delays, simple application problem i.e. integral equation was solved. The examples are not meant to be exhaustive studies of particular application; rather it points out how the time delay is used.

1. Statement of problem

The mathematical description of many problems of engineering interest contains integral equations. Typical of a large class of such problems is the Fredholm integral equation of the second kind, namely

$$y(x) = f(x) + \lambda_{a}^{b} K(x,t)y(t)dt \qquad (1)$$

where f(x) and the Kernel K(x,t) are given functions

a and b are constants

λ is a parameter

y(x) is the unknown solution

The following equation was actually solved:

$$y(x) = \frac{2}{3} + 2_{0}^{1} (x-t)\, y(t)dt \qquad (2)$$

with the initial approximation Yo(t) = 0. The step size was chosen as $\Delta X = 0.025$, i.e. 1/40 and the Fisher's iteration method is used. This equation is a trivial example whose solution is easily seen to be y(x) = x.

2. Method of solution and result

Fig. 3.1 illustrates a two-time-scale hybrid computer method designed to solve a simple integral equation of (2). The integration variable t is represented by "fast" time $\tau = 1250t$ in an ASTRAC II iterative differential analyzer, while x varies in 40 discrete steps

Δx between 0 and 1, stepping once per computer run.

Our time delay system is used as a recirculating function memory (40 DATA words). This memory supplies a complete trial approximation $y_k(t)$ once per computer run, and one function value is erased and corrected at the end of each run (every 1 millisecond of minor cycle). Therefore all function value corrections take 40 millisecond (major cycle).

Fig 3.2 and Fig 3.3 show logic diagram and timing chart for this example. Fig 3.4 presents a flow chart of the solution procedure; a complete listing of the PDP-9 program is given in Table 1.

Fig 3.5 shows successive approximations to the integral equation.

Fig 3.6 shows typical convergence characteristic of this hybrid computer solution with initial approximation $y_o(x) = 0$. Where solutions $y_n(t)$ were compared with exact analytical solution, $Z(t)$ by means of root-mean-square criterion, defined by

$$ERROR_n = \sum_{i=1}^{I} Z(xi) - y_n (Xi)^{2}$$

Fig 3.7 shows the solution wave form of this problem.

ACKNOWLEDGMENTS

The writer is grateful to the office of U. S. National Aeronautics and Space Administration and to Dr. E. W. Scott, U. S. National Research Council for supporting this work under NASA International University Fellowship fund 5606-800-196.

The project was carried out in the Computer Science Research Laboratory, Electrical Engineering Department of the University of Arizona.

The project was suggested and guided by Professor G. A. Korn.

Thanks are also due Dr. R. H. Mattson, Head, Electrical Engineering Department for his encouragement and providing the necessary facilities.

REFERENCES

1. PDP-9 User Handbook. Digital Equipment Corporation.

2. DIGITAL LOGIC Handbook, Digital Equipment Corporation.

3. PDP-9, PDP-9/L INTERFACE Manual, Digital Equipment Corporation.

4. PDP-9 MAINTENANCE MANUAL, Digital Equipment Corporation.

5. PDP-9 MACRO-9 Assembler, Digital Equipment Corporation

6. H. O. Eide and J. A. Forrester; A Hibrid Transport Delay Simulation, The Boeing Company, Attachment A 2-5323-4/77, 1963.

7. Korn, G. A., and T. M. Korn; Electronic Analog and Hybrid Computers, McGraw, 1964.

8. Mackay, D. M., and M. E. Fisher; Analog Computing at Ultra-high speed, Wiley, 1962.

9. Bekey, G. A. and W. J. Karplus; Hybrid Computation, Wiley, 1968.

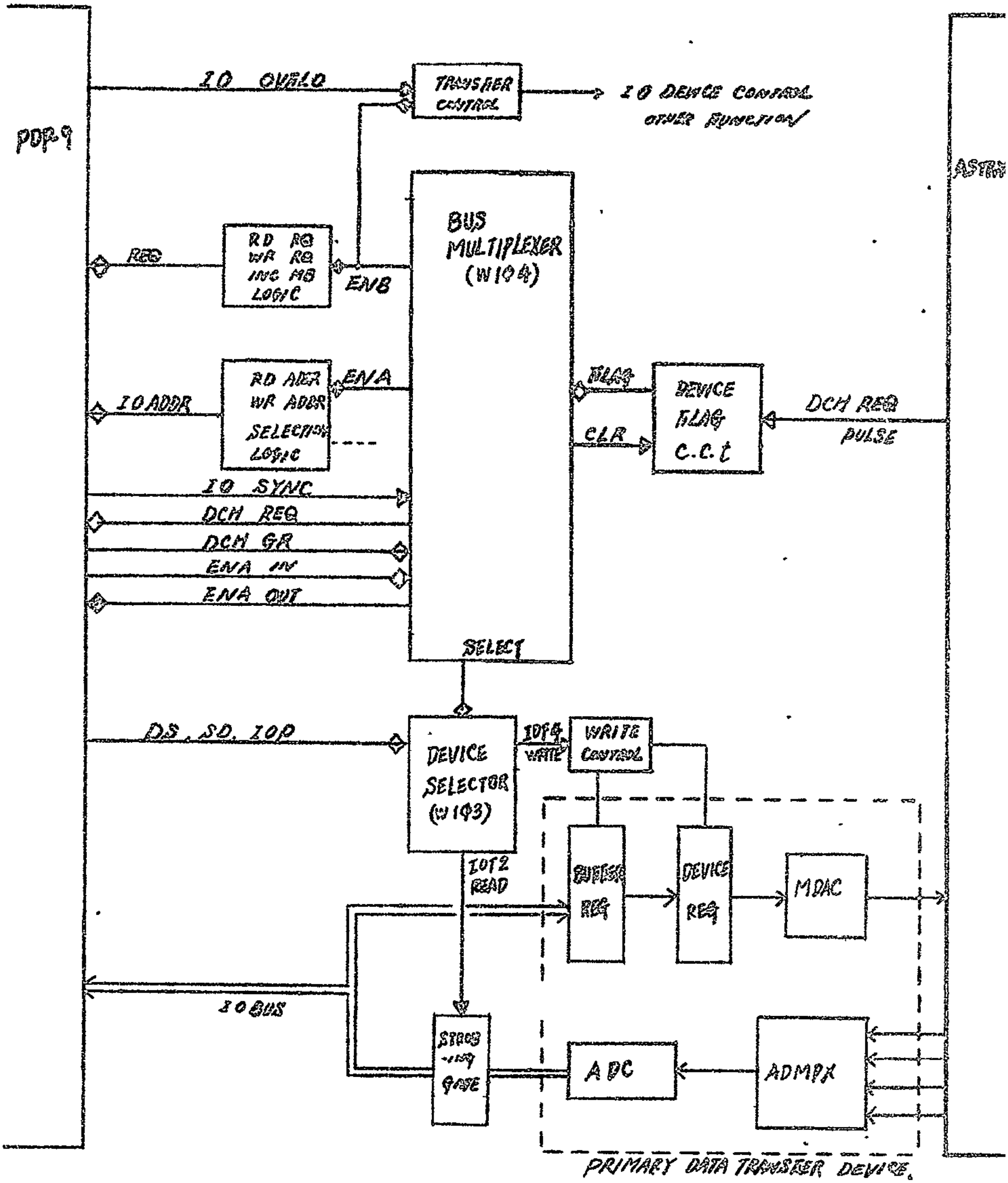


FIG 1.1 BLOCK DIAGRAM OF DATA CHANNEL INTERFACE

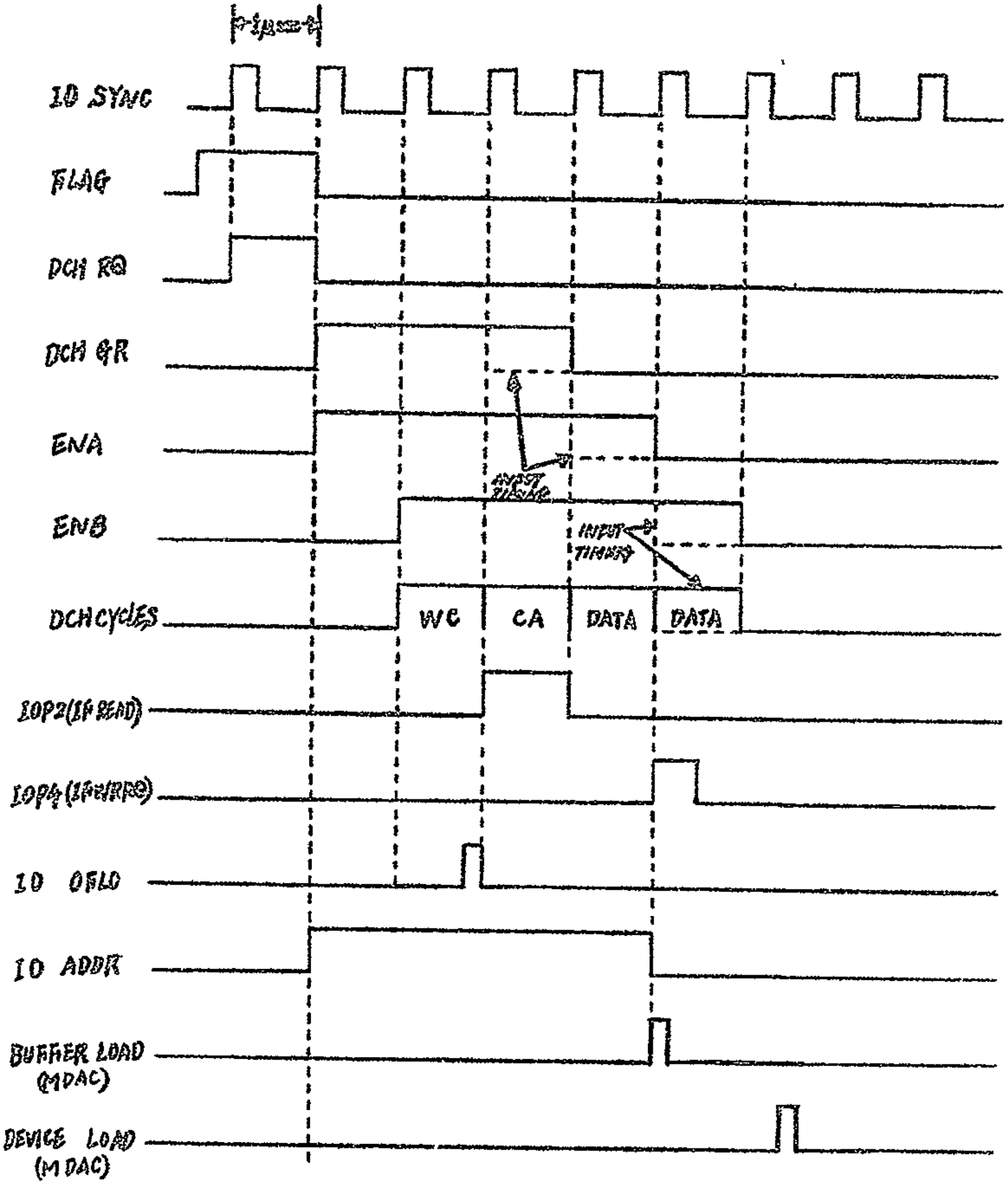
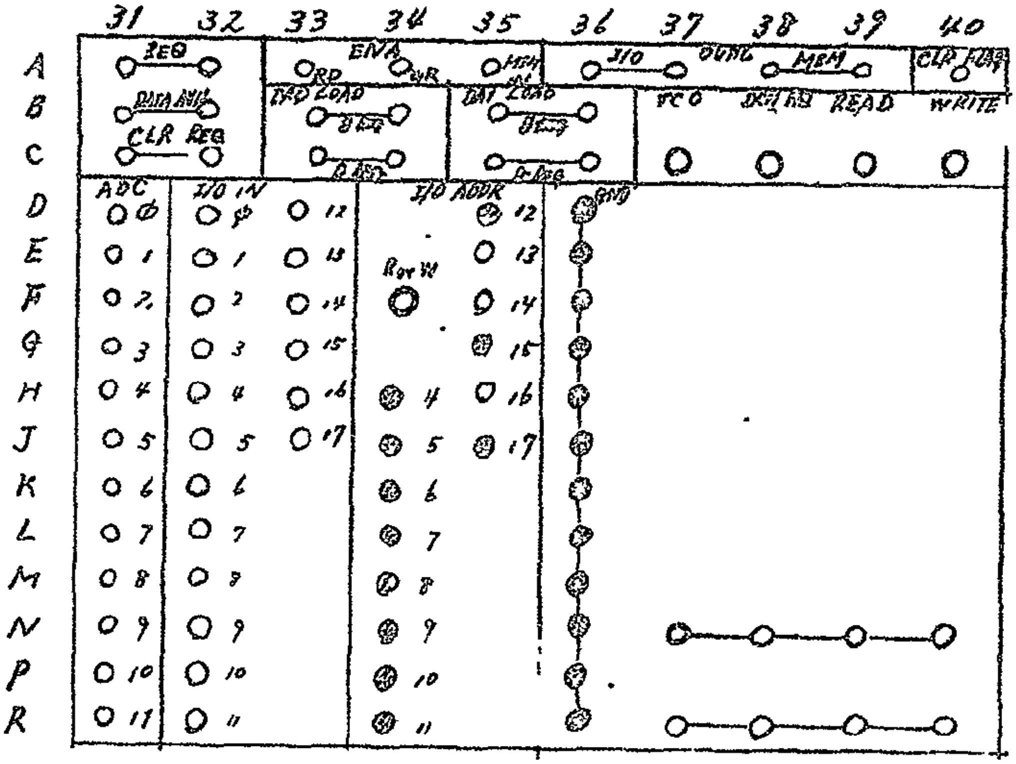
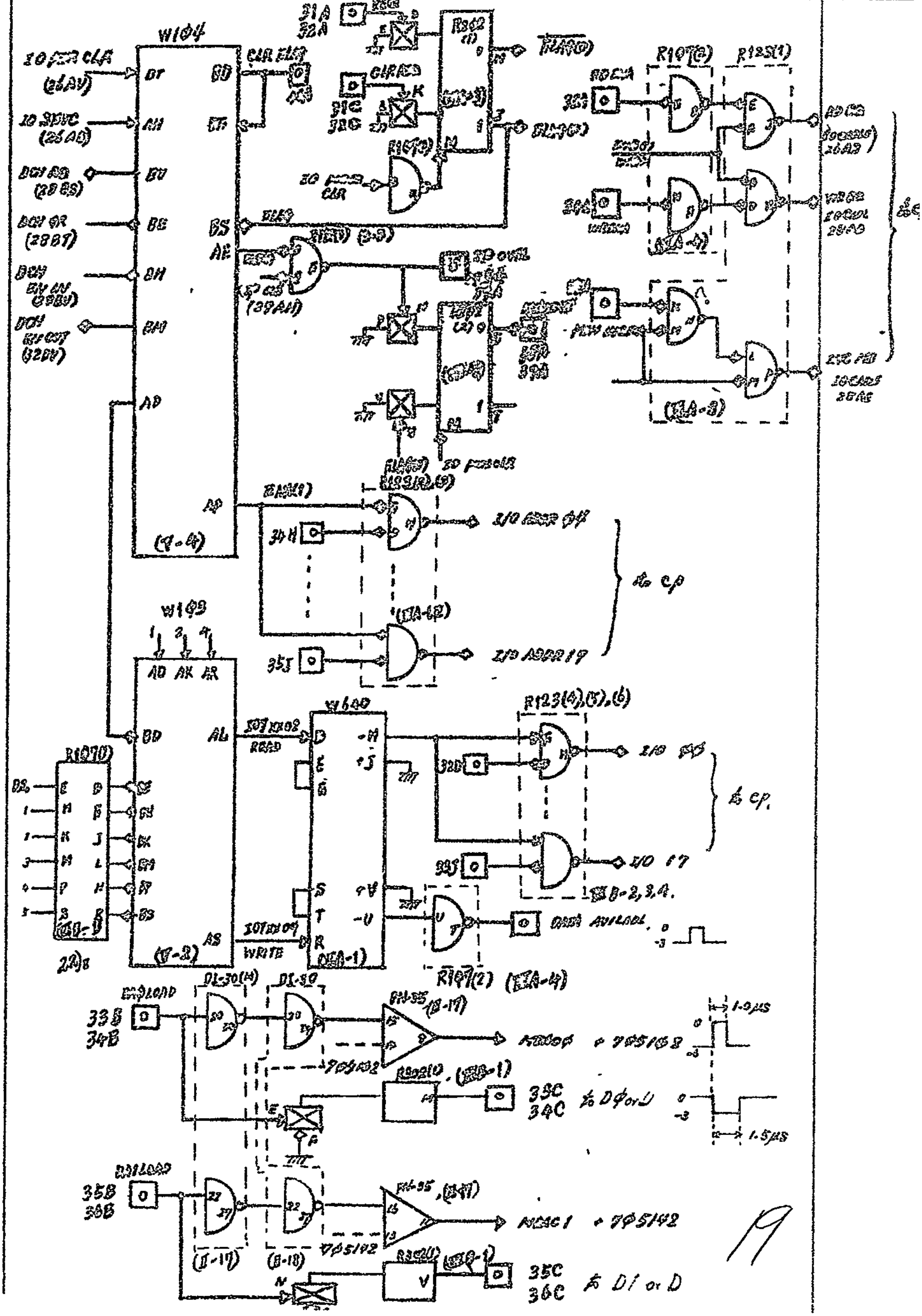


FIG 1.2 DCH TIMING

FIG 1.3 FRONT VIEW OF LPB FOR ASTRAC II/PDP-9 DCH




W104
IO PER CLR (26AV)
IO SYNC (26AD)
DCH RS (28BS)
DCH OR (28BT)
DCH BV IN (29BV)
DCH BV OUT (32DV)
BF
AH
BV
BE
BH
BM
AP
BD
ET
BS
AE
AR
(9-4)
CLR ELM
31A
32A
R902
FLAG
CLR FLG
31C
32C
ELM(0)
IO PWR CLR
R107(2)
R107(8)
R123(1)
RD CMD
33J
WRCMD
RD CB
IOCMDS 26AB
WR CB
IOCMDS 26AD
R107(1) (9-9)
37 OVFL
37 CS (39AH)
EXT
FLW MODE
39A
DSR
ZYC PD
IOCMDS 26AE
(EIA-3)
FLAG(1)
IO PWRCLR
ELM(1)
R123(4),(5)
34H
I/O ADDR 04
to cp
(EIA-12)
35J
I/O ADDR 17
W103
AD AK AR
R107(1)
(EIA-1)
22A
BD
AL
AS
(9-2)
507 R102 READ
507 R109 WRITE
W640
(DEA-1)
R123(4),(5),(6)
32D
I/O F3
to cp.
33J
I/O F7
EIB-2,3,4.
DATA AVAILABLE
0
-3
R107(2) (EIA-4)
DL-30(H)
DL-30
33B
34B
DMLOAD
DN-35 (B-17)
NOC0
705192
1.0µS
R902(1)
(EIB-1)
35C
34C
to D0 or U
1.5µS
35B
36B
DNLOAD
DN-35 (B-17)
NOC1
705192
(B-17)
(B-18)
35C
36C
to D1 or D

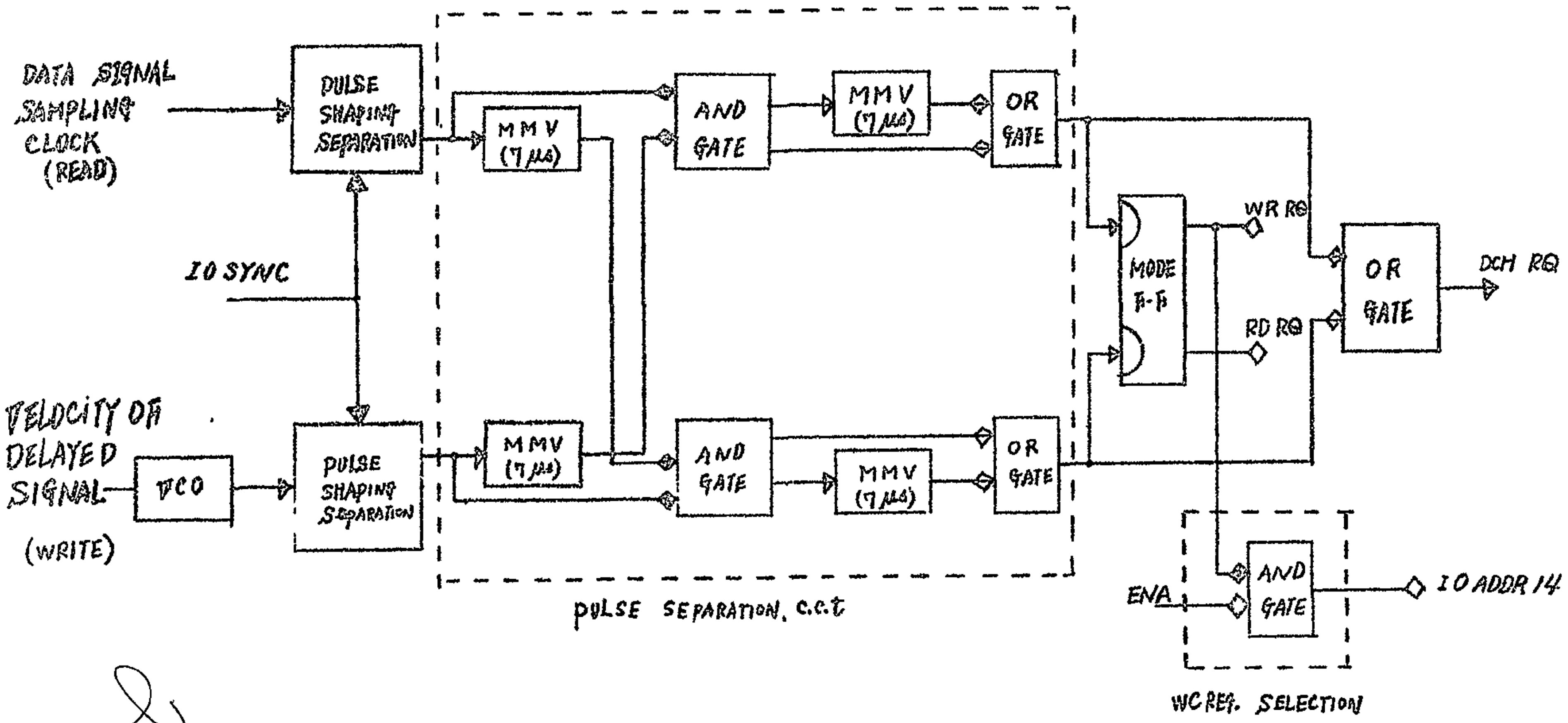
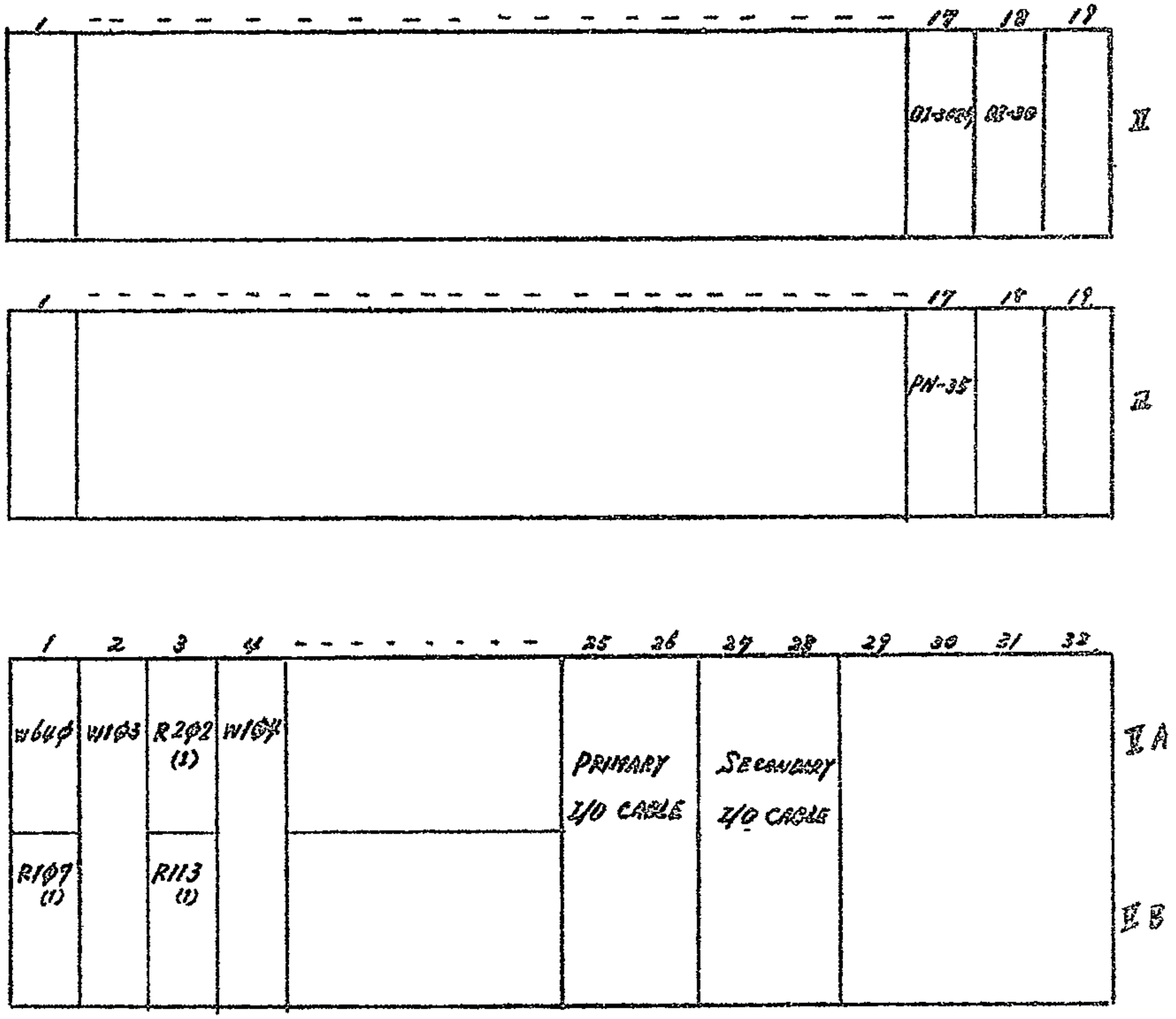


FIG 2·1 VARIABLE TIME DELAY LOGIC FOR DCH READING AND WRITING

FIG 1·5 CARD LAYOUT FOR. RSTPACE/PDP-9 DCH (FRONT VIEW)

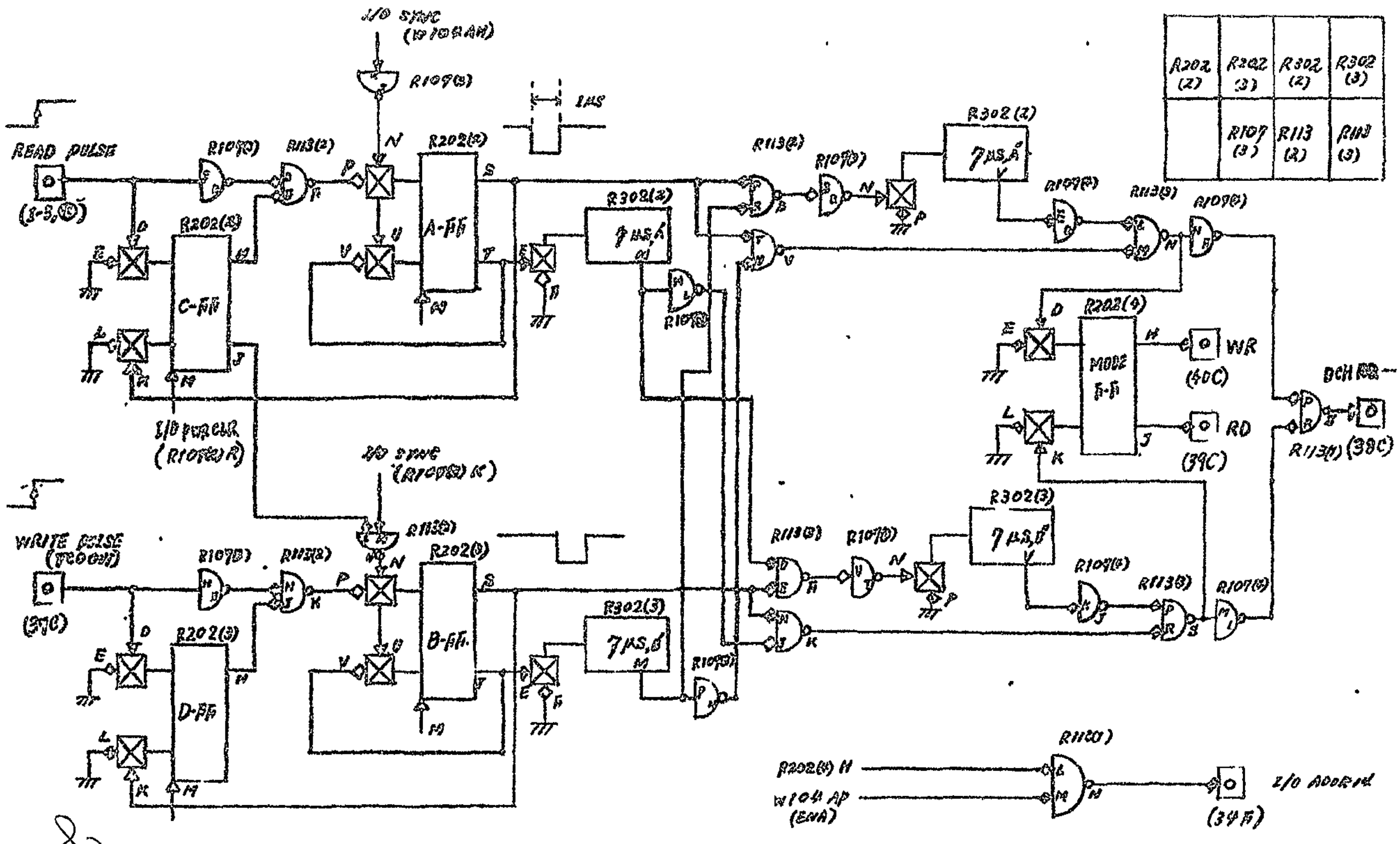


	III A	III B
4	R107(2)	R123(4)
3	R123(1)	R123(5)
2	R123(2)	R123(6)
1	R123(3)	R302(1)
		R202(4)
	R107(4)	

FRONT

R202 (2)	R202 (3)	R302 (2) No Card	R302 (3)
	R107 (3)	R113 (2)	R113 (3)

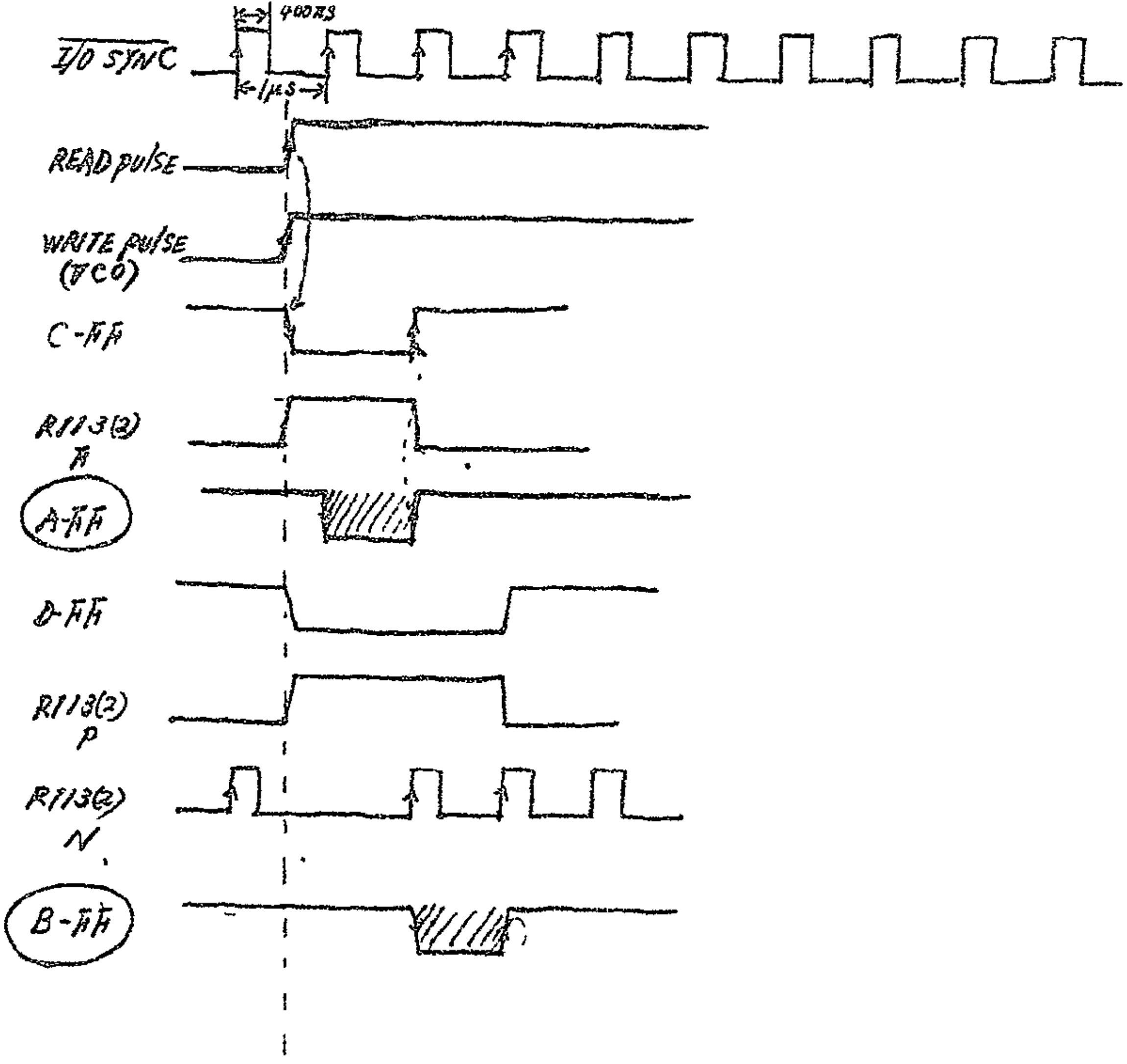
	WD	CA
READ	32	33
WRITE	22	23



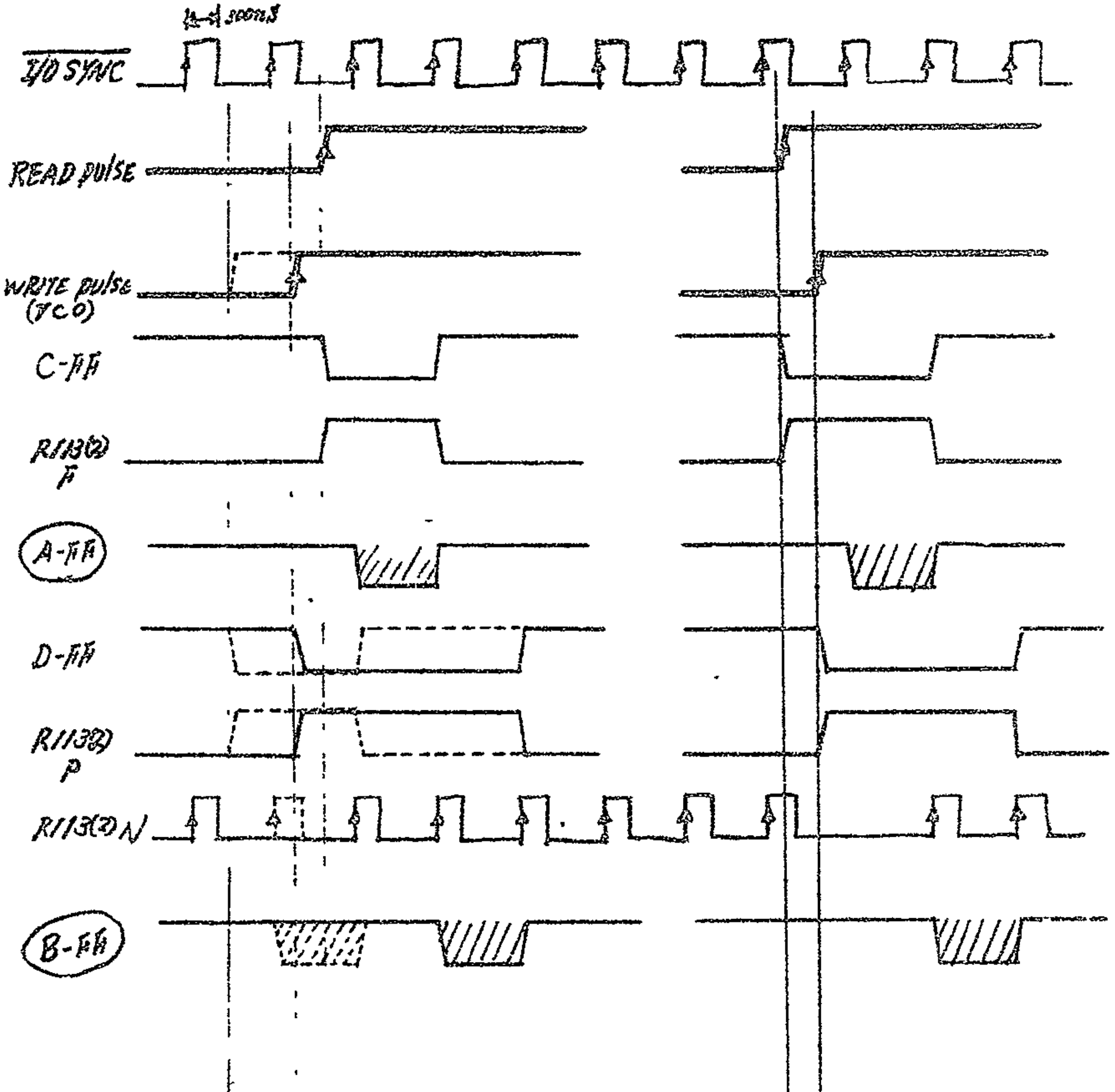
R202 (2)	R202 (3)	R302 (2)	R302 (3)
	R107 (3)	R113 (2)	R113 (3)

FIG 2-8. DETAILED LOGIC C.C.T OF VARIABLE TIME DELAY

FIG 2·3. one clock pulse time phase shift c.c.t. (PULSE SHAPING SEPARATOR C.C.T)



CASE 1 ; READ AND WRITE Request pulse come simultaneously



Case 2; WRITE pulse comes fatster than READ pulse, OR VICEVER

FIG 2.4 PULSE SEPARATION C.C.t (MORE THAN 7μsec)

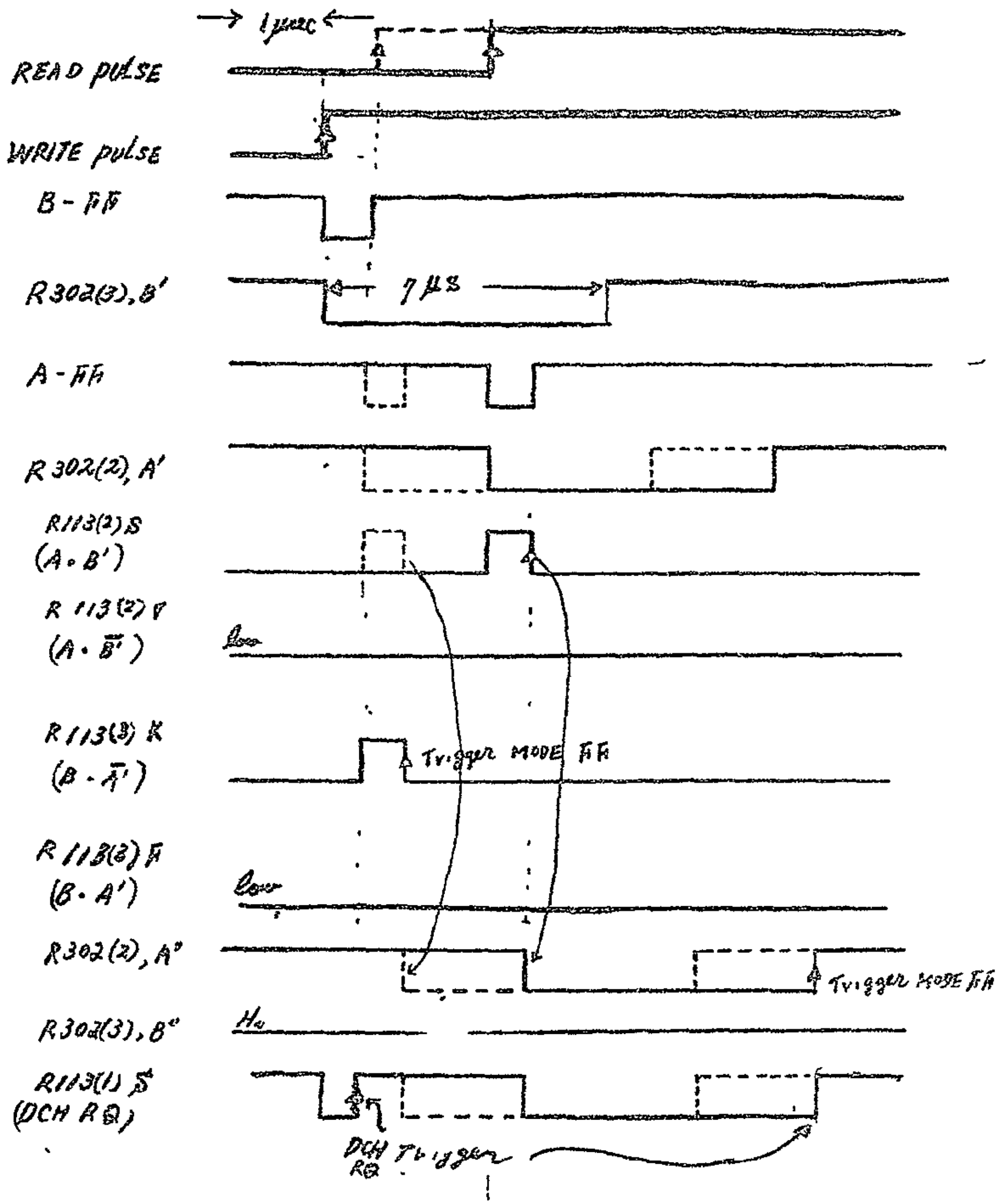
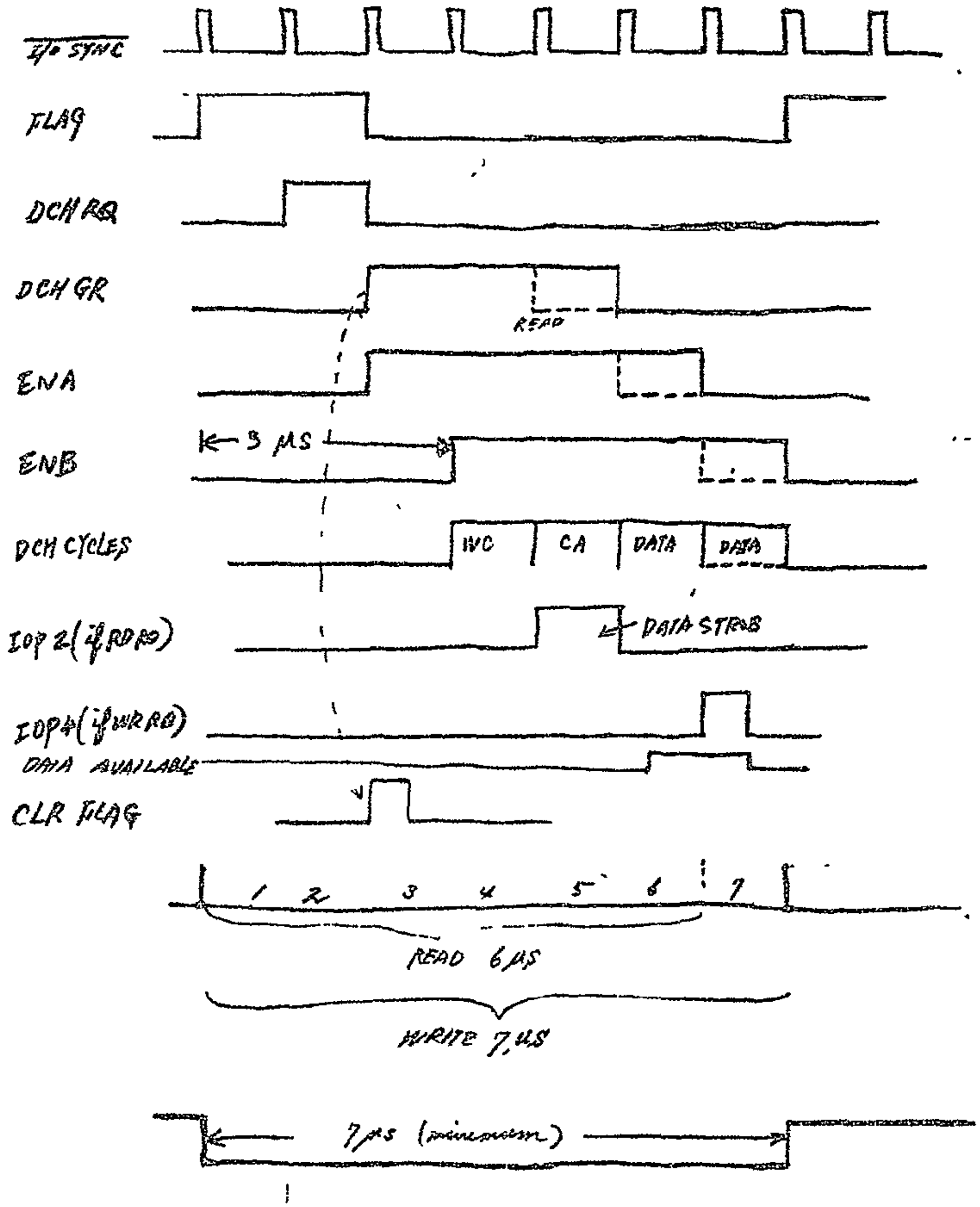


FIG 2.5 <u>Minimum pulse interval between RD pulse and WR pulse.</u>



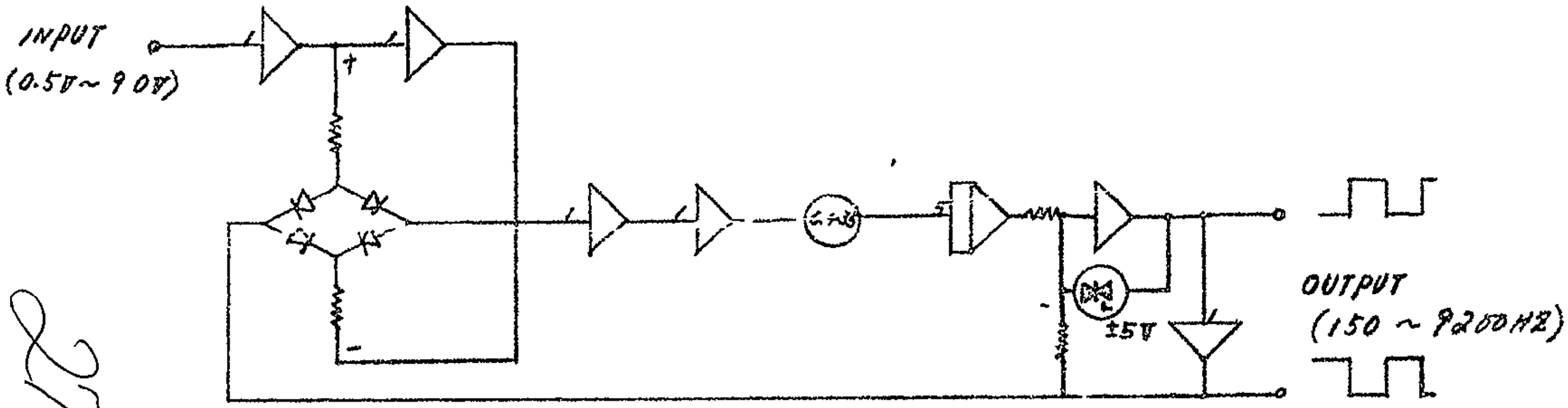


FIG 2.6 ANALOG VOLTAGE CONTROLLED OSCILLATOR (V.C.O)

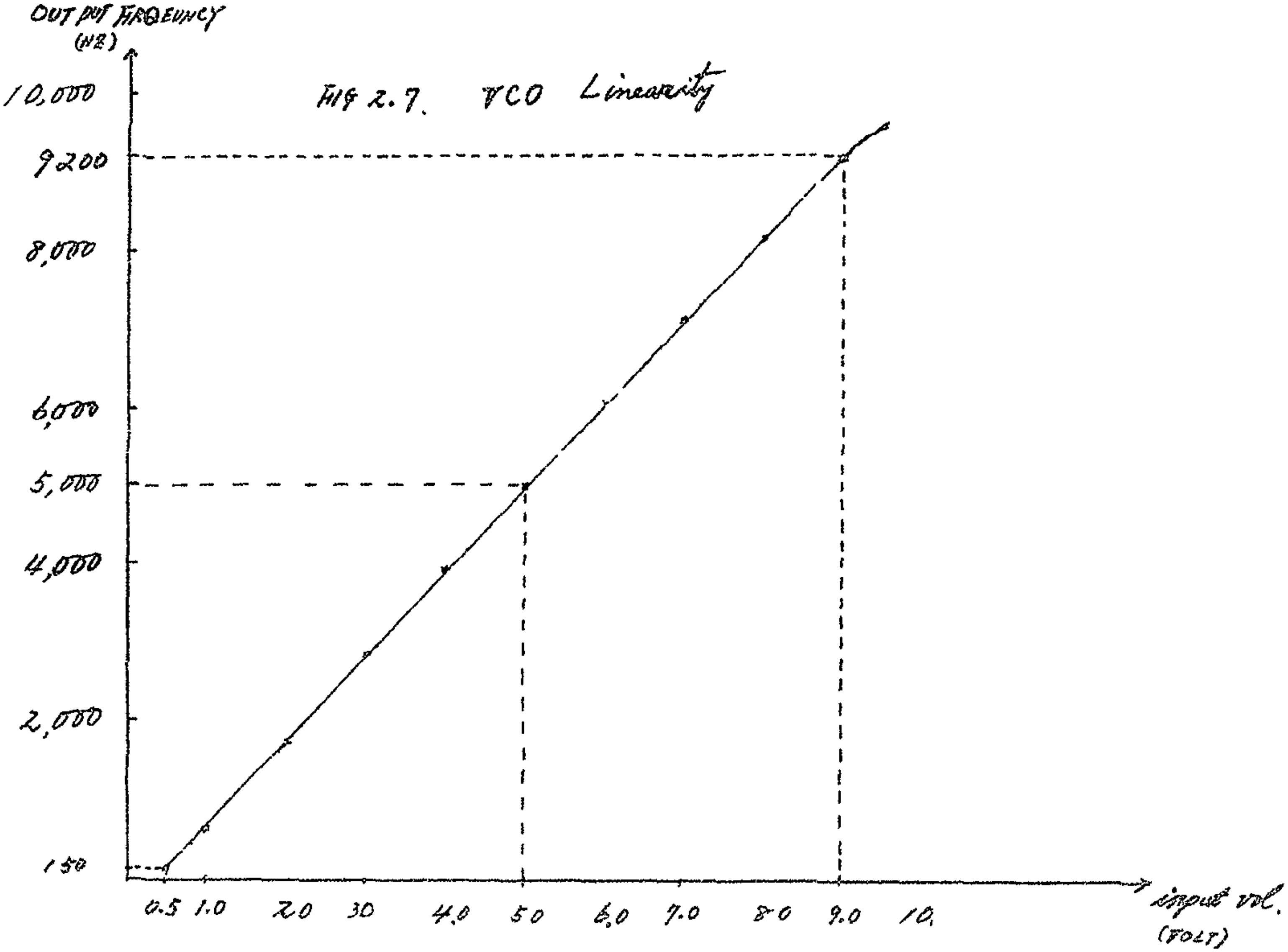
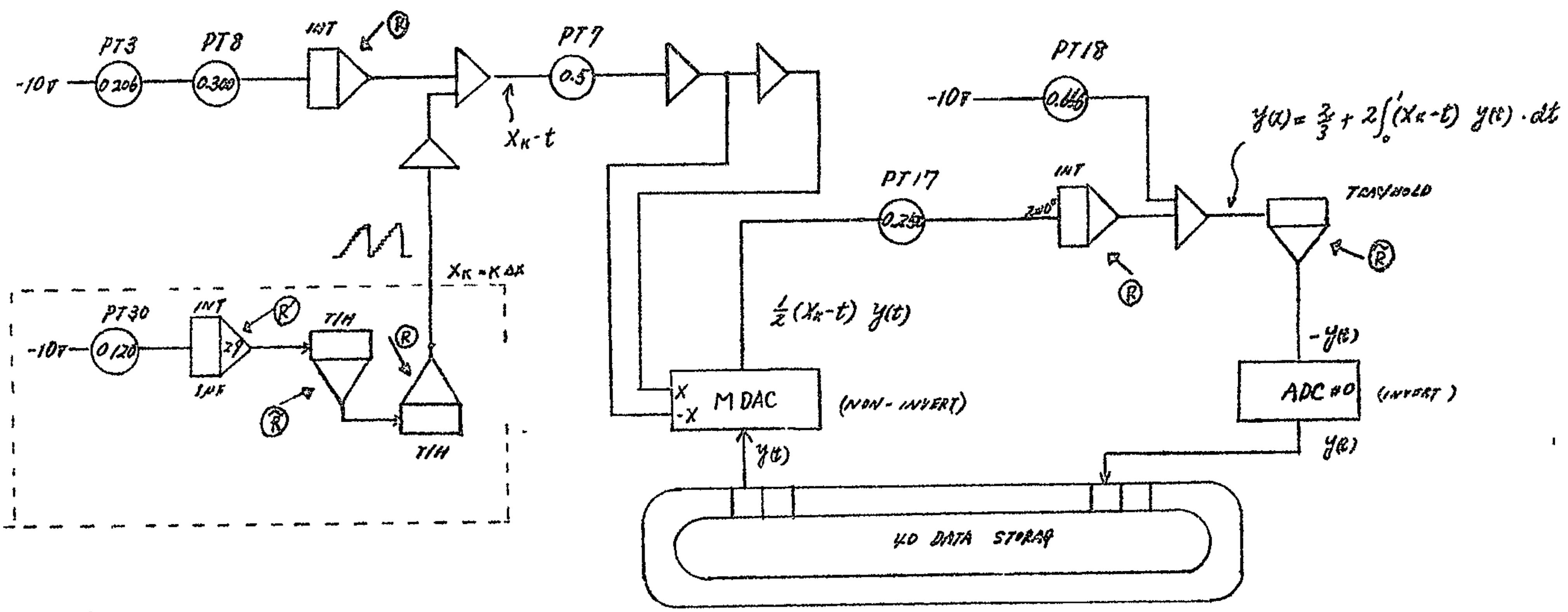


FIG 2.7. VCO Linearity



A/C RATE	1000 /SEC
T	800 μs
t_0	970 μs
n	40 ms
N	50 ms
REPEAT SW	ON

FIG 3.1 Hybrid computer solution of an integral equation, using time delay

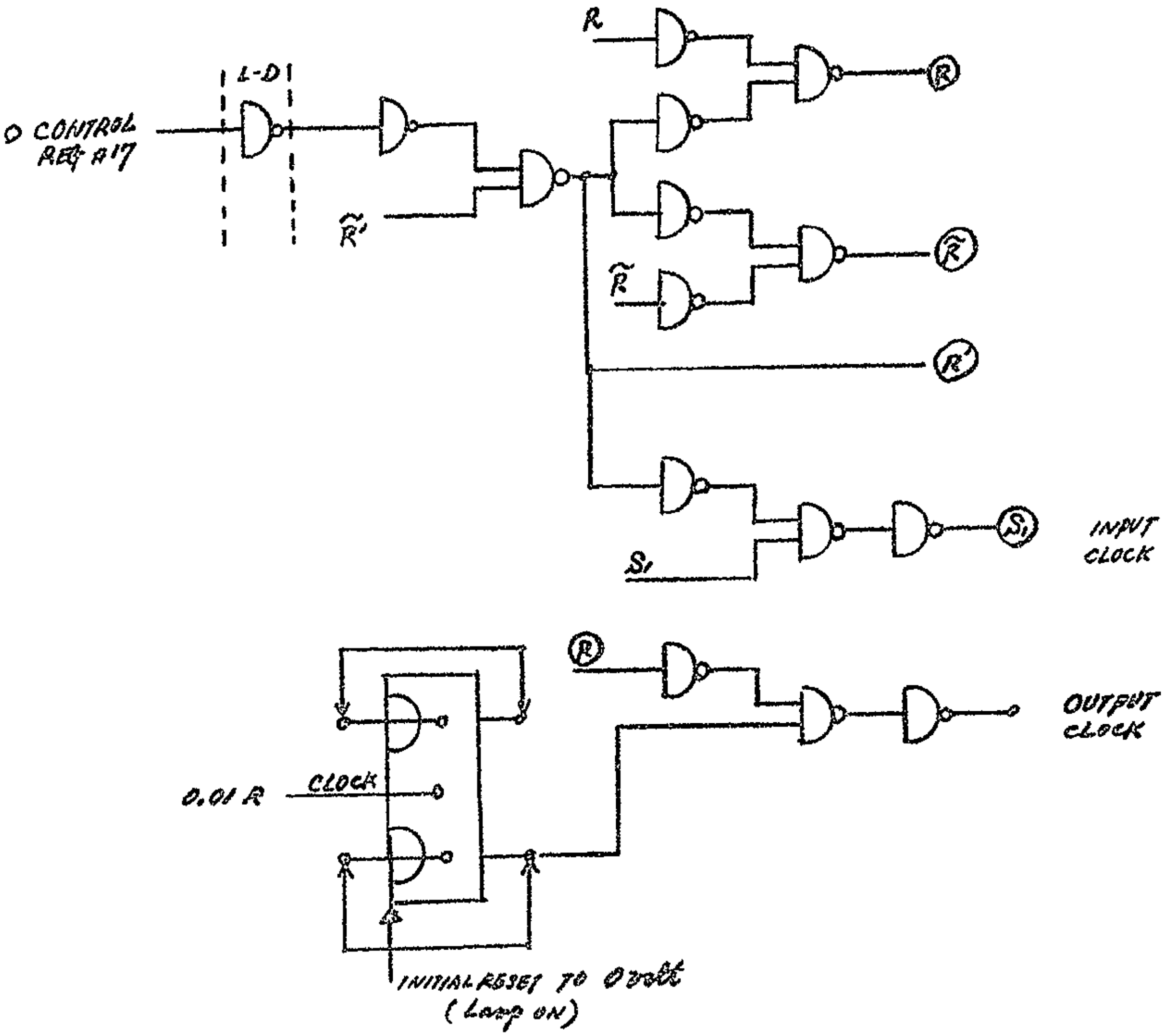


FIG. 3.2 LOGIC DIAGRAM OF TIMING PULSE FOR INTEGRAL EQUATION

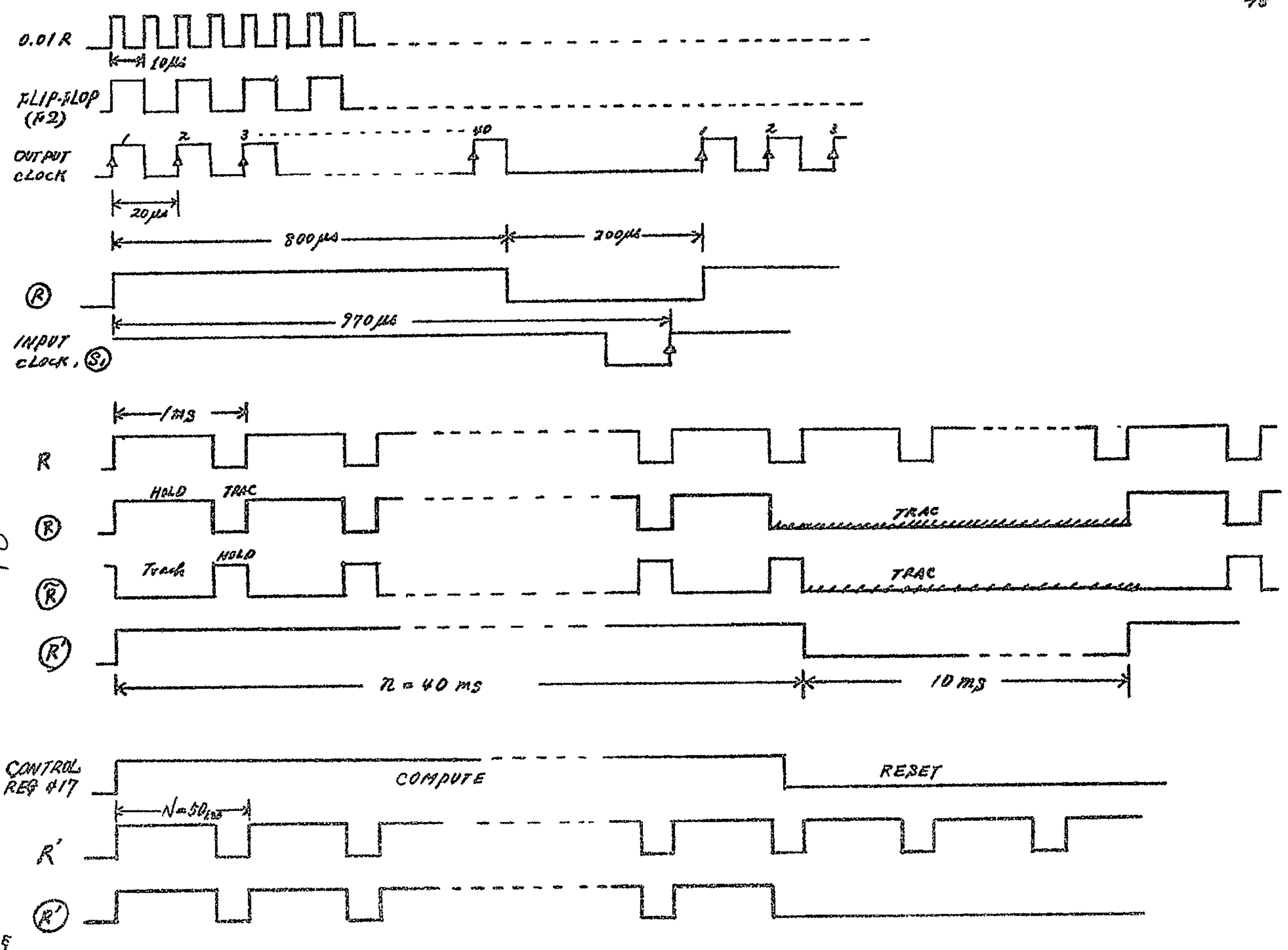
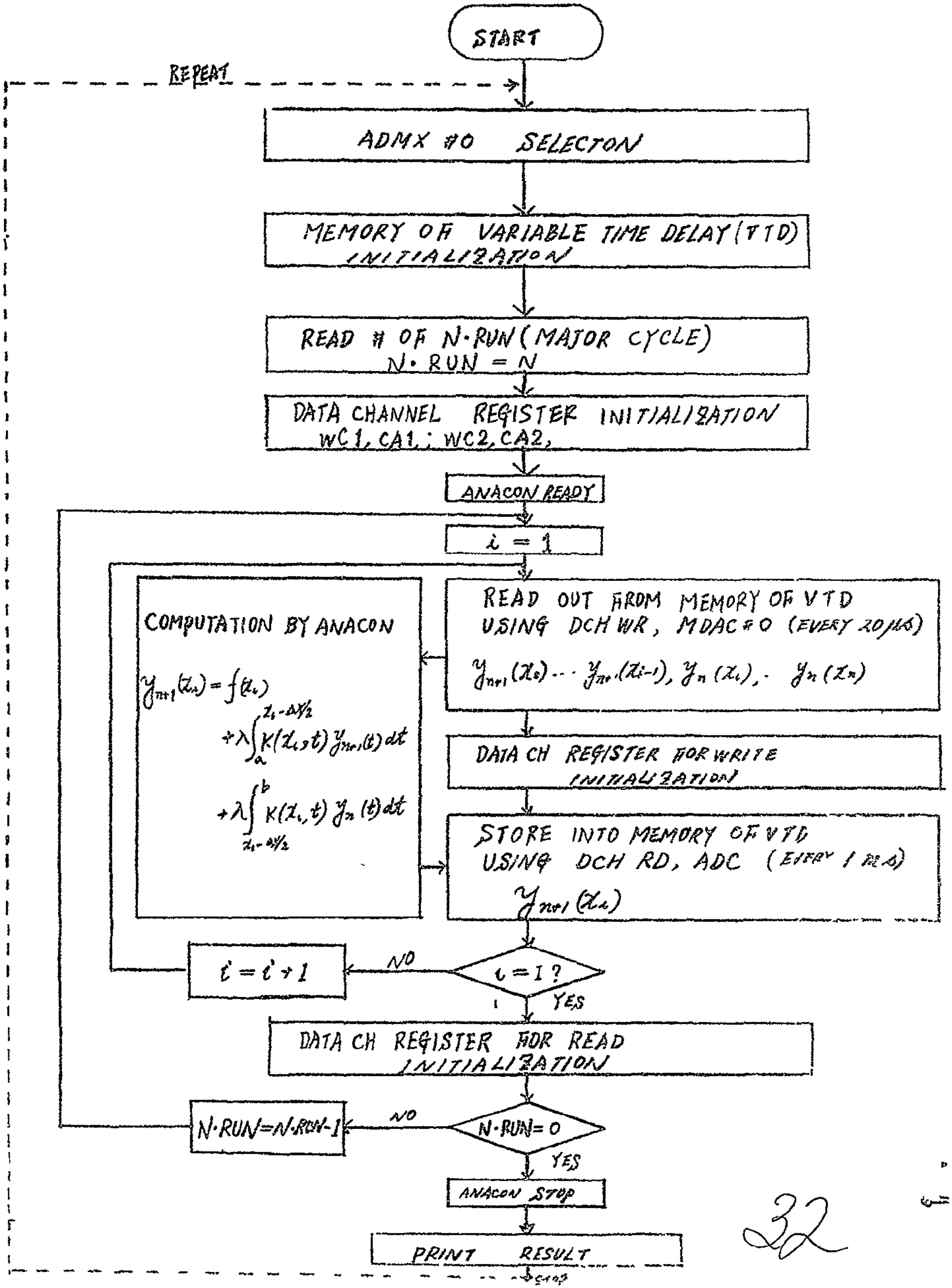


FIG 3-3 TIMING CHART FOR SOLVING INTEGRAL EQUATION

FIG 3 4. HYBRID COMPUTATION FLOW DIAGRAM OF INTEGRAL EQUATION


START
REPEAT
ADMX #0 SELECTON
MEMORY OF VARIABLE TIME DELAY (VTD) INITIALIZATION
READ # OF N·RUN (MAJOR CYCLE)
N·RUN = N
DATA CHANNEL REGISTER INITIALIZATION
WC1, CA1 ; WC2, CA2,
ANACON READY
$i = 1$
COMPUTATION BY ANACON
$y_{n+1}(x_i) = f(x_i) + \lambda \int_a^{x_i - \Delta x/2} K(x_i, t)\, y_{n+1}(t)\, dt + \lambda \int_{x_i - \Delta x/2}^{b} K(x_i, t)\, y_n(t)\, dt$
READ OUT FROM MEMORY OF VTD
USING DCH WR, MDAC #0 (EVERY 20 μs)
$y_{n+1}(x_0) \cdots y_{n+1}(x_{i-1}), y_n(x_i), \cdot\ y_n(x_n)$
DATA CH REGISTER FOR WRITE INITIALIZATION
STORE INTO MEMORY OF VTD
USING DCH RD, ADC (EVERY 1 ms)
$y_{n+1}(x_i)$
$i = I$?
NO
$i = i + 1$
YES
DATA CH REGISTER FOR READ INITIALIZATION
N·RUN = 0
NO
N·RUN=N·RUN-1
YES
ANACON STOP
PRINT RESULT
STOP

DEC 11 '89

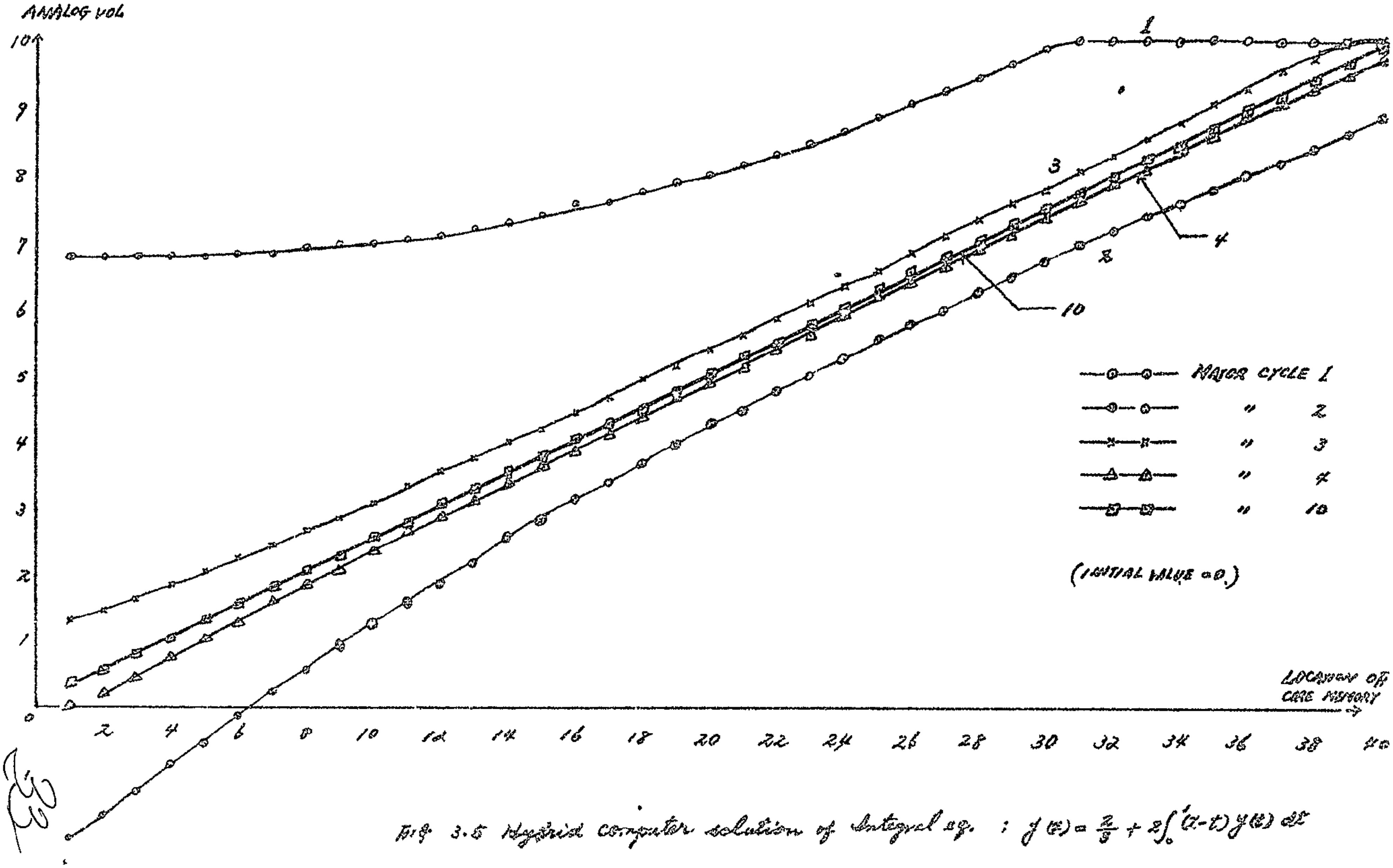


Fig 3.5 Hybrid computer solution of Integral eq. : $y(t) = \frac{2}{3} + 2\int_0^1 (\tau - t)\, y(\tau)\, d\tau$

Dec. 17 '69

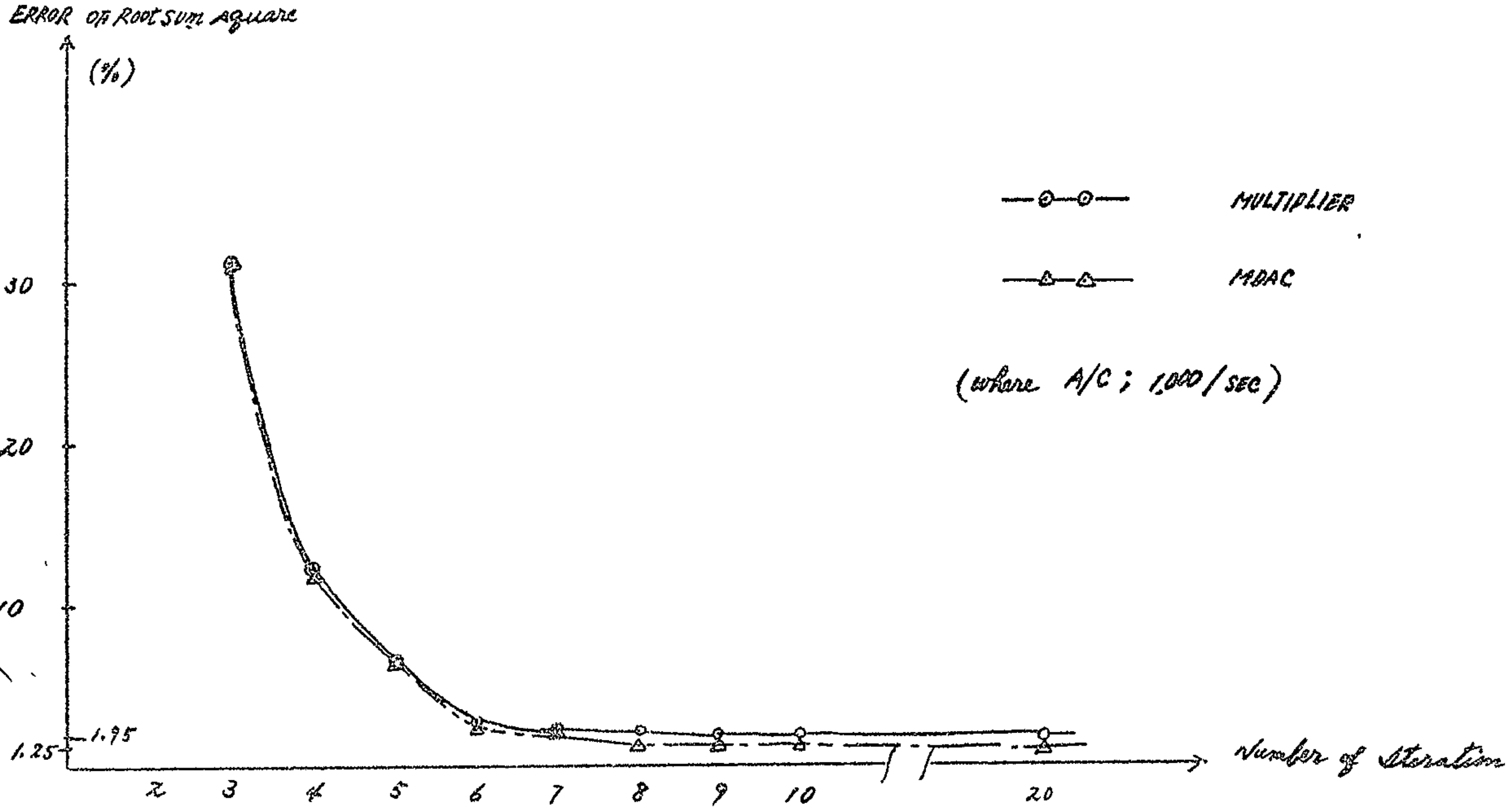


FIG 3.6 CONVERGENCE CHARACTERISTIC

```
                                   /NAKA NOV28,1969
                                   /TITLE VARIABL TIME DELAYS(VTD)
                                   /       USING DATA CHANNEL INTERFACE
                                   /EXAMPL: SOLUTION OF INTEGRAL EQUATION
                                   /         USING VTD
                                   /
                                          .GLOBL OUTPUT,INPUT
                                   /SELECTION OF ADMX #Ø
ØØØØØ   R  75ØØØØ  A               START  CLA             /LOAD AC WITE ZERO
 ØØØØ1  R  7Ø5Ø24  A                       7Ø5Ø24          /SELECT ADMX #ØØ
                                   /MEMORY INITIALIZATOIN
                                   /
ØØØØ2   R  2ØØØ17  R                      LAC    WC
ØØØØ3   R  Ø4ØØ22  R                      DAC    N.RUN
ØØØØ4   R  2ØØØ2Ø  R                      LAC    CA1
ØØØØ5   R  Ø6Ø151  R                      DAC*   (1Ø     /LOC.1Ø=CA1
ØØØØ6   R  2ØØØ16  R                      LAC    IDATA   /LOAD AC WITH ZERO
ØØØØ7   R  Ø6ØØ1Ø  A               CLEAR  DAC*   1Ø      /SET IDATA
ØØØ1Ø   R  44ØØ22  R                      ISZ    N.RUN   /RECORD # OF MEMORY
                                                         /TO BE ERASED
ØØØ11   R  6ØØØØ7  R                      JMP    CLEAR
                                   /
                                   /READ # OF N.RUN(MAJOR CYCLE
ØØØ12   R  12Ø147  E                      JMS*   INPUT
ØØØ13   R  6ØØØ15  R                      JMP    .+2
ØØØ14   R  ØØØØ23  R                      CONSTØ
ØØØ15   R  6ØØØ25  R                      JMP    MAINØ   /GO TO MAIN
                                   /
                                   /TABLE OF CONSTANT
                                   /
ØØØ16   R  ØØØØØØ  A               IDATA  ØØØØØØ         /INITIAL DAT@
ØØØ17   R  77773Ø  A               WC     77773Ø         /# OF WORDS TO BE
                                                         /TRANSFERD(2,COMPL)
ØØØ2Ø   R  ØØØØ76  R               CA1    A-1            /STARTING ADDR FOR
                                                         /INPUT
ØØØ21   R  ØØØØ76  R               CA2    A-1            /START ADDR FOR OUT
ØØØ22   R  ØØØØØØ  A               N.RUN  Ø              /WORKING STRAGE FOR
                                                         /# OF N.RUN
ØØØ23   R  ØØØØØØ  A               CONSTØ Ø              /# OF N.RUN SET BY
                                                         /INPUT ROUTIN
ØØØ24   R  ØØØØØ1  A               CONST1 ØØØØØ1         /USED BY COTRL REG
                                   /
ØØØ25   R  2ØØØ17  R               MAINØ  LAC    WC      /DCH REGISTER
ØØØ26   R  Ø6Ø152  R                      DAC*   (22     /INITIALIZATION
ØØØ27   R  Ø6Ø153  R                      DAC*   (32     /ROUTIN
ØØØ3Ø   R  2ØØØ2Ø  R                      LAC    CA1
ØØØ31   R  Ø6Ø154  R                      DAC*   (33
ØØØ32   R  2ØØØ21  R                      LAC    CA2
ØØØ33   R  Ø6Ø155  R                      DAC*   (23
ØØØ34   R  2ØØØ23  R                      LAC    CONSTØ
ØØØ35   R  Ø4ØØ22  R                      DAC    N.RUN   /# OF MAJOR CYCLE
ØØØ36   R  2ØØØ24  R                      LAC    CONST1
ØØØ37   R  7Ø52Ø4  A                      ACRL           /LOAD CONTRLREG 1
                                   /
ØØØ4Ø   R  74ØØØØ  A               MAIN1  NOP            /DATA TRANSFER
```

```
      ØØØ41  R  74ØØØØ  A               NOP             /VIA DATA CHANNEL
      ØØØ42  R  22Ø152  R               LAC*    (22     /DCH IN REG WC2
      ØØØ43  R  74Ø2ØØ  A               SZA             /IS ZERO?
      ØØØ44  R  6ØØØ46  R               JMP     .+2
      ØØØ45  R  6ØØØ57  R               JMP     SR2     /GO TO WC2 SERVICE
                                                        /ROUTIN
      ØØØ46  R  22Ø153  R       MAIN2   LAC*    (32     /DCH OUT REG WC1
      ØØØ47  R  74Ø2ØØ  A               SZA             /IS ZERO?
      ØØØ5Ø  R  6ØØØ4Ø  R               JMP     MAIN1   /RETURN TO MAIN
      ØØØ51  R  6ØØØ52  R               JMP     SR1     /GO TO WC1 SERVICE
                                                        /ROUTIN
                                /
                                /
      ØØØ52  R  2ØØØ17  R       SR1     LAC     WC      /DCH OUT REG WC1,CA
       ØØØ53  R  Ø6Ø153  R               DAC*    (32     /1 INITI ROUTIN
      ØØØ54  R  2ØØØ2Ø  R               LAC     CA1
      ØØØ55  R  Ø6Ø154  R               DAC*    (33
      ØØØ56  R  6ØØØ4Ø  R               JMP     MAIN1   /RETURN TO MAIN
                                /
      ØØØ57  R  2ØØØ17  R       SR2     LAC     WC      /DCH IN REG WC2,CA
      ØØØ6Ø  R  Ø6Ø152  R               DAC*    (22     /2 INITI ROUTIN
      ØØØ61  R  2ØØØ21  R               LAC     CA2
      ØØØ62  R  Ø6Ø155  R               DAC*    (23
      ØØØ63  R  44ØØ22  R               ISZ     N.RUN   /RECORD # OF MJORE
                                                        /CYCLE
      ØØØ64  R  6ØØØ46  R               JMP     MAIN2   /RETURN TO MAIN
      ØØØ65  R  75ØØØØ  A               CLA             /LOAD AC WITH ZEROS
      ØØØ66  R  7Ø52Ø4  A               ACRL            /TURN COTRL REG OFF
                                 /TYPE OUT COMPUTATION RESULT USING F4
      ØØØ67  R  12Ø15Ø  E               JMS*    OUTPUT
      ØØØ7Ø  R  6ØØØ72  R               JMP     .+2
      ØØØ71  R  ØØØØ76  R               DATA
      ØØØ72  R  6ØØØØØ  R               JMP     START
                                /ARRAY DESCRIPTOR BLOCK
      ØØØ73  R  ØØØØ5Ø  A               ØØØØ5Ø          /4Ø(DCMAL) OF WORDS
      ØØØ74  R  ØØØØØØ  A               ØØØØØØ          /Ø FOR 1-D ARRAY
      ØØØ75  R  ØØØØØØ  A               ØØØØØØ          /Ø FOR 1-D ARRAY
      ØØØ76  R  ØØØØ77  R       DATA    A               /ADDR OF IRT DATA
      ØØØ77  R          A       A       .BLOCK 5Ø       /4Ø DATA WORDS
                ØØØØØØ  R               .END    START
      ØØ147  R  ØØØ147  E *ETV
      ØØ15Ø  R  ØØØ15Ø  E *ETV
      ØØ151  R  ØØØØ1Ø  A *LIT
      ØØ152  R  ØØØØ22  A *LIT
      ØØ153  R  ØØØØ32  A *LIT
      ØØ154  R  ØØØØ33  A *LIT
      ØØ155  R  ØØØØ23  A *LIT
                                          NO ERROR LINES


MACRO UA44

>^C
```

```
C NAKA1 DCE.3,1969
C PRINT 40 DAT@
C
      SUBROUTINE OUTPUT(K)
      DIMENSION     K(40),XK(40)
      WRITE(5,100)
C CONVERTION INTO ANALOG-VOLTAGE
C 2.EXP(17)-1=131071
      DO 10 I=1,40
10    XK(I)=FLOAT(K(I))/13107.1
      WRITE(5,101) (I,XK(I),I=1,40)
C COMPUTATION OF ROOT SUM SQUARE ERROR
C
      SUM=0
      DO 20 I=1,40
20    SUM=SUM+(10./40.*FLOAT(I)
    1 -XK(I))**2
C ROOT SUM SQUARE ERROR IN PERCENT
      ERROR=10.*SQRT(SUM)
      WRITE(5,102) ERROR
      RETURN
100   FORMAT(1X,*DATA IN MEMORY*)
101   FORMAT(1H,I5,5X,F10.3)
102   FORMAT(1X,*ERROR OF ROOT SUM SQUA
    1RE=*,F10.4,*%*)
      END

FORTR@N 4 UA5A
>B, L_NAKA2


END PASS1

C NAKA2 DEC.3,1969
C READ A NUMBER OF MAJOR CYCLE
C
      SUBROUTINE INPUT(N)
      WRITE(5,200)
      READ(4,201)    N
      RETURN
200   FORMAT(1X,*READ # OF MAJOR CYCLE
    1 IN I7*)
201   FORMAT(I7)
      END

FORTRAN 4 UA5A
>
```

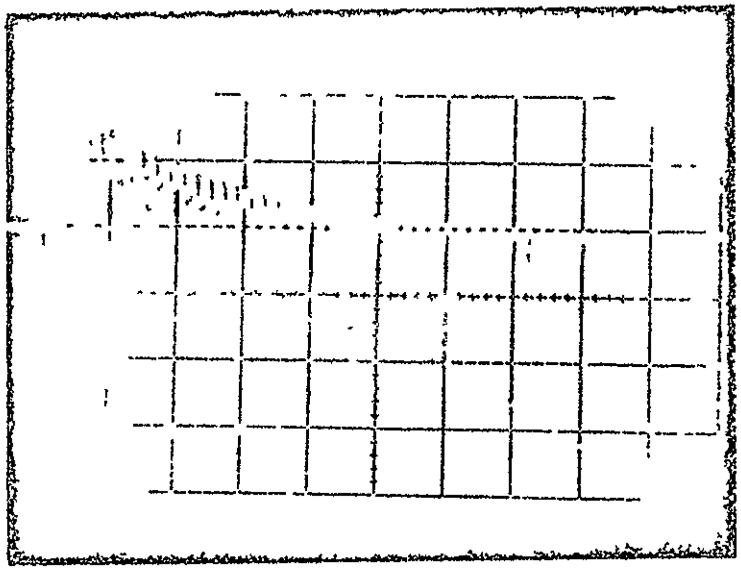


FIG 3.8 OUTPUT WAVE FORM OF TRAC/HOLD AMP. (- y(t))

(5 msec/cm , 2 Volt/cm)

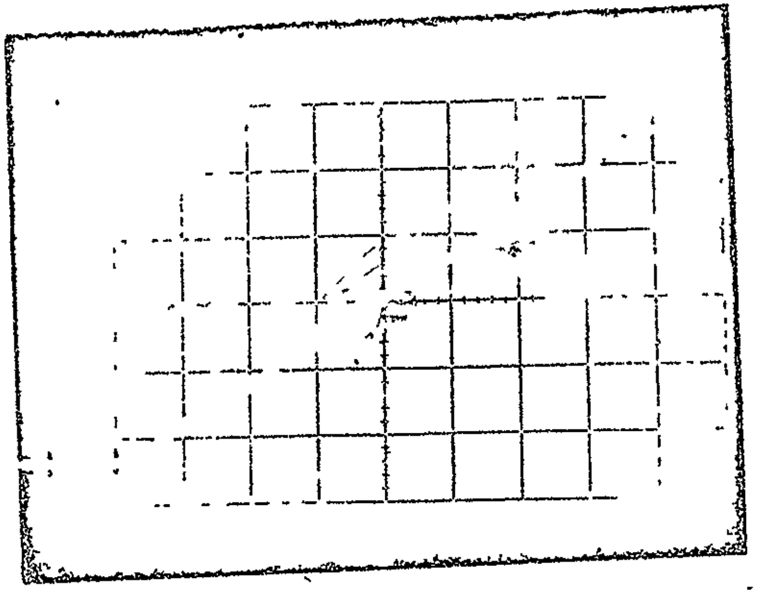


FIG 3.7 Solution; $\tilde{y}(x) = x$.

(5 m sec/cm, 2 volt/cm.)