6/21/61 BELVIN

PATSI — A Block Diagram Compiler for TX-2

Programming Aid To System Investigation

PATSI is a TX-2 compiler for system simulation. The simulation of a system — composed of elements like filters, adders, multipliers, gates, delays and the like — can be easily programmed from the block diagram of the system. A useful feature is the ability to connect a scope to any or all of the waveforms in the system, and observe the progress of the simulation. The user may also interact with his system through the use of various knobs and switches on the TX-2 console.

PATSI has been used in the simulation of several speech-compression devices, and has considerably lessened the burden of programming them.

A typical PATSI statement (one line of typing) describes one block or element in the block diagram. It must give the element a name, or tag; it must tell what type of element the block is; and it must specify the parameters of the block, such as the input(s), gain, frequencies, etc.

```
TAG → ELEMENT TYPE | PARAMETERS
```

For example,

```
Z → ADDER | X, Y, W13
```

describes an element called Z whose output is the sum of its inputs. These inputs are the outputs of the elements called X, Y, and W13.

The tag serves three purposes. It identifies the line in the PATSI program, it serves as the name of the element, and it is the name of the output of the element.

For a given element type, i.e., COSINE GENERATOR, ADDER, DELAY, the form of the statement, i.e., the order and meaning of the parameters, is found in the "DICTIONARY," along with the limitations and restrictions of the particular block. The dictionary form of ADDER, for example, is

```
ADDER | in1, in2, in3, ....
```

The last line of every program is "DONE."

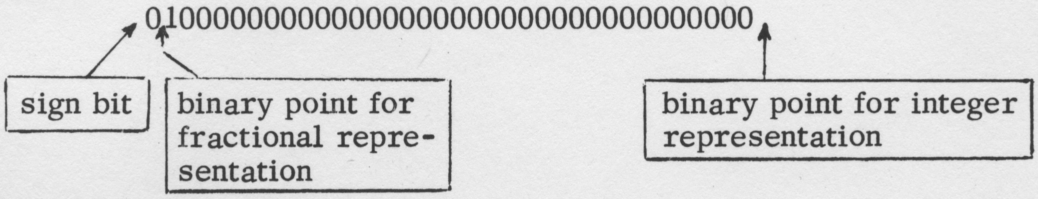
The following points should be kept in mind in the specification of parameters as numbers:

1. A number typed will be treated as base 8 (instead of base 10) unless followed by a period. Thus, when we mean a number to be decimal system, as we usually

do, it is followed by a period.

2. A number typed is treated as an integer unless preceeded by a period. Mixed numbers are not allowed.

3. A collection of bits in a register has no intrinsic numerical value until the position of the decimal point (binary point) is specified. The binary point position must be specified or assumed before a collection of bits can be called equal to a number. In rule 2, "treated as" an integer means that the binary point in the register is taken to be after the least significant bit. "Treated as" a fraction means the binary point is considered to be in front of the most significant bit. Thus, typing .5. results in the same collection of bits as typing 17179869184. (which is 2^{34}). The binary representation of each is



4. Once a number has been represented in a register as a collection of bits, the location of the binary point is lost. However, each element will deal with parameters as if the decimal point were where it is expected. Thus, GAIN expects an integer, whereas ATTENUATE expects a fraction. Where the DICTIONARY does not suggest a specific form for the parameter, the element probably can operate on the number in the register without needing to know where the binary point is. In this case, it will assume that the binary point in the parameter register is in the same position as the binary point in the waveform register. For example, CLIP | in,.5. > .25. is exactly equivalent to

CLIP | in, ~~.~~17179869184.> 8589934592.

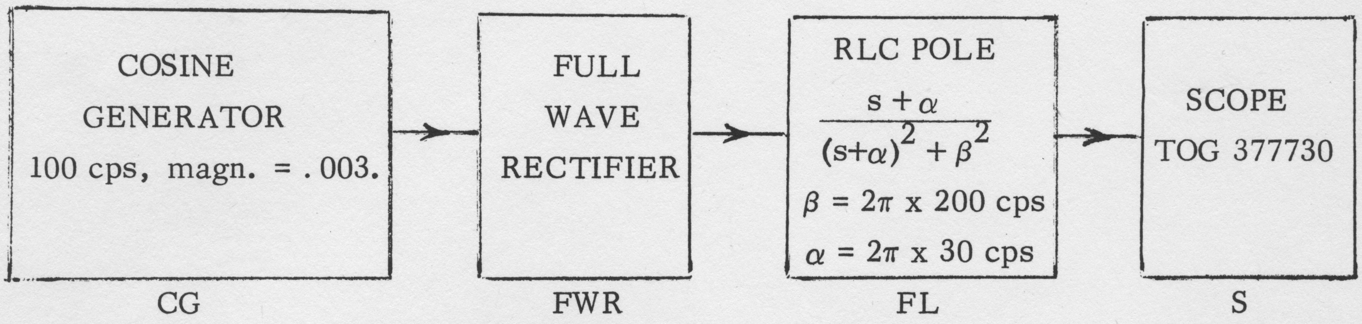
5. Register addresses are most easily specified as octal integers (as 377723).

Lights and Meanings

When the system simulated runs into trouble such as machine overflow, or when the system does something which results in an impossible situation, such as presenting the square root taker with a negative input, certain lights may be lit up to warn the user. Only the lights mentioned below are used. Other troubles may not be detected.

Light Number	Meaning
1	Output has filled core region allotted to it.
2	Modulator output has clipped.
3	Gain output has clipped.
4	Divider output has clipped.
5	Tried square root of negative quantity.
6	Not used.
7	Overflow for adder or difference.
8	Not used.
9	Not used.

Example:



Let the sampling rate be 10000/sec. The PATSI program is:

```
CG → COSINE GENERATOR | .01., .003.
FWR → FULL WAVE RECTIFIER | CG
FL → RLC POLE | FWR, .003., .02.
S → SCOPE | FL, 377730
     DONE
```

ORDER OF COMPILED PROGRAM SAME AS ORDER OF SYMBOLIC PROGRAM

PATSI Dictionary

I. Elements - the number in parenthesis is the page on which a description of the element may be found.

(9)	ADDER
(9)	ATTENUATE
(10)	BAND PASS FILTER
(11)	CLIP
(12)	CONTROLLED OSCILLATOR (square wave)
(10)	CONVOLVE
(7)	COSINE GENERATOR
(9)	DELAY
(9)	DIFFERENCE
(11)	DIVIDER
(10)	EXPOSINE ($h(t) = e^{-\alpha t} \sin \beta t$; $H(S) = \frac{\beta}{(s+\alpha)^2 + \beta^2}$)
(11)	FULL WAVE RECTIFIER
(9)	GAIN
(11)	GATE
(11)	GREATER OF
(11)	HALF WAVE RECTIFIER
(7)	INPUT (from core memory)
(9)	INVERTER
(11)	LIMITER
(11)	MODULATOR (multiplier)
(7)	NOISE GENERATOR
(12)	ONE SHOT (monostable multivibrator)
(8)	OUTPUT (to core memory)
(11)	PEAK DETECTOR
(7)	PERIODIC INPUT (from core memory)
(10)	RC POLE ($h(t) = \alpha e^{-\alpha t}$; $H(S) = \frac{\alpha}{s + \alpha}$)
(10)	RLC POLE ($h(t) = e^{-\alpha t} \cos \beta t$; $H(S) = \frac{s + \alpha}{(s+\alpha)^2 + \beta^2}$)
(10)	RLC ZERO ($H(S) = \frac{(s+\alpha)^2 + \beta^2}{\beta}$)
(12)	SAMPLE AND HOLD

(8)	SCOPE
(9)	SCOPE SYNC
(12)	SQUARE ROOT
(12)	SWITCH
(7)	VARIABLE RATE PULSER (hand controlled)
(8)	XYSCOPE
(11)	ZERO CROSSING PULSER

II. Control Statements - these are not elements, although they may relate to them. The parenthsis again contain page numbers.

(13)	BRANCH UNLESS	
(13)	BREAK	
(7)	CHANGE COSINE FREQUENCY	
(10)	CHANGE EXPOSINE	for moveable poles and zeros
(10)	CHANGE RC POLE	
(10)	CHANGE RLC POLE	
(10)	CHANGE RLC ZERO	
(13)	MULTIPLY T BY	
(13)	RETURN	

3 3 DEFINE

3 3 DEF → ELEMENT | in1, in2, ...

X → Output Element | inj, ink, ...

Y → other Element | in 0, in m, ...

⋮

3 3 END

INPUT | $\alpha \rightarrow \beta$

use S-memory for Input/Outp
up to 130,000

The output of this block at successive sampling times is the contents of successive registers of core memory.

α = first } Register of the area of core memory containing the desired
β = last } input waveform.

If the data is to be configured, the number of the configuration is superscripted after β, e.g., INPUT | $1 \rightarrow 100000^{12}$

When $\alpha \rightarrow \beta$ is used up (after (β-α) samples), we go to MKIV.

PERIODIC INPUT | $\alpha \rightarrow \beta$

This differs from the above only in that when $\alpha \rightarrow \beta$ is used up, we return to α. Thus the waveform in $\alpha \rightarrow \beta$ is repeated endlessly.

COSINE GENERATOR | F, M

F = cosine frequency as a fraction of the sampling frequency.
M = amplitude. It should be large, or the difference equation used will not be effective.
F should be greater than .00003.

The output of this block is a sample of $\cos 2\pi F/F_s t$. The corresponding sine wave ($\sin 2\pi F/F_s t$) is found at X + 2 where X is the tag of the cosine generator.

NOISE GENERATOR

No parameters. The output is a pseudorandom uniformly distributed noise sequence.

VARIABLE RATE PULSER | M

The output is normally 0^-. It is replaced with a one sample pulse of amplitude M periodically. The repetition rate is given by the contents of the left half of the KNOB.

$$\text{pulse frequency} = \frac{\text{KNOB}_{3,4}}{4\ 000\ 000} \times \text{sampling frequency.}$$

— really $\frac{\text{KNOB}_{3,4}}{2^{19}} \doteq \frac{\text{KNOB}_{3,4}}{\frac{1}{2}\times 10^6}$?

CHANGE COSINE FREQUENCY | X, ω

X is the tag of the cosine generator affected. ω is the tag of an element whose output is to control the frequency of X. The output of ω is taken as a fraction of the sampling frequency.

This operation is slow and should be done only once per several sampling times if possible.

OUTPUT | in, $\alpha \rightarrow \beta$

This element allows the waveform at "in" to be saved in core memory. α and β are as for INPUT. Configuration is allowed. α and β must be in S memory ($\alpha < \beta < 200000$). When $\alpha \rightarrow \beta$ is used up, new samples are ignored, and push button 1 lights up.

SCOPE | in, CT

in = tag of element whose output is fed to the scope.
CT = control toggle, used as follows:

$$4.10 \quad \begin{cases} 0 & \text{show waveform} \\ 1 & \text{do not show waveform} \end{cases}$$

Q3 - sweep frequency. The number of samples corresponding to the scope face is given by $\frac{100000_8}{Q3}$.

Q2, Q4 are amplitude controls. Gain of scope = . (Q2) x $2^{(Q4)}$.
Q1 = vertical position control.

If several scopes are used, all the sweep rates are added together. Good practice is to set all but one to zero so that sweep rate control is by only one toggle register. (See scope sync.)

XYSCOPE | x, Cx, y, Cy

x = tag of horizontal input.
y = tag of vertical input.
Cx = Control toggle for horizontal input, Cy = control toggle for vertical input, as follows:

4.10 as for SCOPE
Q2, Q4 as for SCOPE
Q1 = horizontal position control for Cx, vertical position control for Cy.
Q3 - not used.

SCOPE SYNC | X > CT

The output of X, and control toggle CT are used to control the scope as follows:

$$4.10 \text{ of CT} = \begin{cases} 0 & \text{no sync} \\ 1 & \text{see below} \end{cases}$$

If the scope trace has not reached the end of the screen nothing happens. If the scope trace has reached the end of the screen, the scope is turned off until the output of X is greater than the number in CT. When this happens, the scope is turned on again, and the trace reset to the left side of the scope face.

DELAY | in, N

The output of this element is the same as the output of "in" but delayed by N sampling intervals. N is a positive integer. DELAY uses some memory from 215777 down, as necessary.

GAIN | in, M

The output is the same as the output of "in" but with a gain of M. M is an integer, positive, negative, or zero. If the product is too big, ± 377777777777 is used, and push button 3 is lit.

ATTENUATE | in, k

Like GAIN, but k is a fraction, positive, negative, or zero.
The combination of GAIN and ATTENUATE can give any fixed multiplier.

INVERTER | in

A gain of -1.

ADDER | in1, in2, in3, ...

The output is the sum of the input waveforms. Overflow is detected and indicated by lighting push button #7, but not corrected. The number of inputs is limited to 12.

DIFFERENCE | in1, in2

The output is (in1) - (in2). Overflow is as for ADDER.

RC POLE | in, F

Sampled data equivalent of $H(S) = \frac{\alpha}{S + \alpha}$. Here F is α divided by the sampling frequency.

RLC POLE | in, F, G

Sampled data equivalent of $H(S) = \frac{S + \alpha}{(S+\alpha)^2 + \beta^2}$. Here

F is α divided by the sampling frequency.
G is β divided by the sampling frequency.

EXPOSINE | in, F, G

Sampled data equivalent of $H(S) = \frac{\beta}{(S+\alpha)^2 + \beta^2}$ α and β related to F, G as in RLC POLE.

RLC ZERO | in, F, G

Sampled data equivalent of $H(S) = \frac{(S+\alpha)^2 + \beta^2}{\beta}$ α, β related to F, G as in RLC POLE.

CONVOLVE | in, $\alpha \rightarrow \beta$

α is the first and β is the last register of an area of core memory containing the function to be convolved with the waveform at "in." This is brute force simulation of filters, and is very slow. Some memory is used from 215777 down as needed.

BAND PASS FILTER | in, F1, G1, F2, G2, ...

A cascade of up to 6 RLC Poles. In addition to the theoretical delay, there is a delay of one sampling interval for each RLC POLE used after the first.

CHANGE RLC POLE | X, α, β
CHANGE EXPOSINE | X, α, β
CHANGE RLC ZERO | X, α, β
CHANGE RC POLE | X, α

These permit moveable poles and zeros. α and β are tags of control elements, and X is the tag of the element whose poles or zeros are to be moved. The output of α corresponds to F and the output of β corresponds to G.

These are slow, and should be used in conjunction with "MULTIPLY T BY | N" when possible.

GATE | in, C in

"in" is the waveform to be gated. C in is the control waveform.

$$\text{output} = \begin{cases} 0^- & \text{Cin} \le 0 \\ (\text{in}) & \text{Cin} > 0 \end{cases}$$

GREATER OF | in1, in2

The output is the greater of the two inputs.

HALF WAVE RECTIFIER | in
FULL WAVE RECTIFIER | in

These are self-explanatory. The HALF WAVE output is 0^- for negative input.

ZERO CROSSING PULSER | in, M

The output is 0^- except following a zero crossing when it is M. The direction of the zero crossing is not noted.

PEAK DETECTOR | in, M

The output is normally 0^+. Following a positive peak (∧), the output is a pulse of height M. Following a negative peak (∨), the output is a pulse of height -M.

LIMITER | in, M

$$\text{output} = \begin{cases} M, \text{ in} \ge 0^+ \\ -M, \text{ in} \le 0^- \end{cases}$$

CLIP | in, T > B

$$\text{output} = \left\{ \begin{array}{l} T, \text{ in} \ge T \\ \text{in}, T \ge \text{in} \ge B \\ B, B \ge \text{in} \end{array} \right\}$$

T and B are fixed levels, with T > B.

MODULATOR | in1 x in2
DIVIDER | in1/in2

The output of MODULATOR is the product of the inputs, scaled appropriately. The product of fractions is scaled 17. places to the left. The product of integers are scaled 19. places to the right. DIVIDER is the inverse, so a ratio of fractions is scaled 17. places to the right, and a ratio of integers is scaled 19. places to the left. Outputs which are too large are replaced by ± 377777777777 and push button #2 for the modulator or #4 for the divider are lit.

SQUARE ROOT | in

The input is treated as a fraction. Thus $\sqrt{in} \geq in$. If (in) is negative, the square root of the magnitude is taken, and push button #5 is lit.

SWITCH | Cin, POSin, NEGin

$$\text{The output is taken from} \begin{cases} \text{POSin} & \text{Cin} \geq 0^{+} \\ \text{NEGin} & \text{Cin} \geq 0^{-} \end{cases}$$

ONE SHOT | Sin, M, ONTIME

This has two states. In the off state, the output and next state are:

$$\begin{cases} 0^{-} \text{ and off if Sin} \leq 0 \\ \text{M and on if Sin} > 0 \end{cases}$$

ONTIME $\leq 2^{17}$?

In the on state, the output and next state are:

$$\begin{cases} \text{M and off if the ON state has lasted "ONTIME" sampling intervals.} \\ \text{M and on if the ON state has not lasted "ONTIME" sampling intervals.} \end{cases}$$

ONTIME is an integer greater than 0.

SAMPLE AND HOLD | in, Cin

$$\text{Output} = \begin{cases} \text{(in)}, & \text{Cin} > 0 \\ \text{last output}, & \text{Cin} \leq 0 \end{cases}$$

CONTROLLED OSCILLATOR | in, M

The output is a square wave of amplitude M and frequency given by the waveform at "in," according to the formula

$$f = \frac{.(in)}{2} \times \text{sampling frequency}$$

```
BREAK
  .
  .
  .
RETURN
```

These are used together. A conventional program placed in between will be executed once each time around the loop.

```
MULTIPLY T BY|N
```

The program or elements following this line are changed or performed only once every N times around the loop. Thus, they seem to have a sampling rate of $\frac{F_s}{N}$, or a sampling interval of NT. N is a positive integer. Sampling rate can not be divided subsequently. See Below

These may be nested so that program parts following two of these statements are looked at every MxN times around the loop.

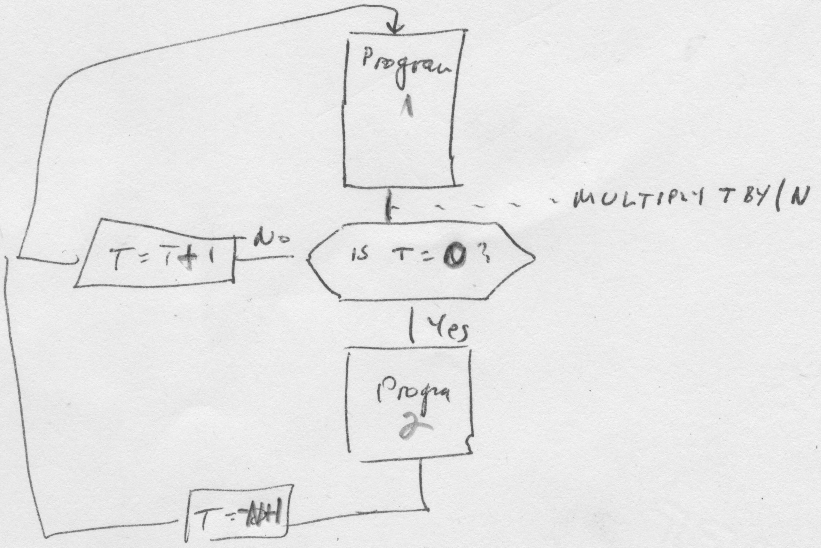
```
BRANCH UNLESS | F = α, → X
```

The simulation will interrupt the normal sequence of control and proceed to line X, except if the output of F is equal to α.

For example, to show a dashed line representation of a function on the scope:

```
MX → COSINE GENERATOR | F, .1
         BRANCH UNLESS | X = 100, → X
         SCOPE | Y, CT
X  →  LIMITER | MX, 100
```

PS - This assumes that "sweep rate" is given by some other scope which is looked at every time around the loop.



∴ goes thru Program 2 first time.

PRINT #4

```
☛☛REP   STARTΔ→SSΔγ
STARTΔ→1STE BACKΔ        **SAVE MARK 4 RETURN ADDRESS
       RFD60ΔBEGIN
☛☛INS   SKIPNSEQΔ
       REX47INPUTΔγ     **INITIALIZE SEQ 47
       IOS4730000       **CONNECT MISC INPUTS
☛☛INS   DSALΔγ
INPUTΔγ→      STE P     **SEQ 47 START POINT. SAVE E.
       hTSD XXΔγ
       3STE #+1
1IOS7530000    **SET LIGHTS = INPUT BUTTON
SKZ1.9XXΔγ     **BUTTON 9 MEANS QUIT
       JMP QUITΔγ
       SKZ1.8XXΔγ       **BUTTON 8 MEANS PRINT, IF MACRO USED
DUMPΔγ→       JMP #+1   **RESET BY DUMPΔ MACRO IF USED
       LDE P
       JPD INPUTΔγ      **RETURN TO PATSI
QUITΔγ→       IOS6020000           **DISCONNECT SCOPE AND LOWER FLAG
       IOS6040000
QUIT1Δ→       RFD70#+1
       REX47INPUTΔγ     **RESET SEQ 47 COUNTER
BACKΔ→        JPQ ##    **RETURN TO MARK 4
☛☛END
```

33 REP   CONTINUE+4   αβγ→   RETURN

FRE OGIX PATSI ~~[illegible]~~ Listing 6/64

```
ΔORG/
 STARTΔ→
 SSΔγ→
 NOSYNCΔγ→

33 RC     FULLSCOPΔγ                  2000.24
          0                                25
          SCOPΔ                            26
          CON 1                            51
          1                                52
          CON 4                            53
          CON 5                            54
          604 353 33 0333                  55
          -43,                             56
          -1,                              57
          268486364154.                    60
          SWGEN Δγ                         61
          LIMIT Δγ                        117
          OUADΔγ                          124
          17.,                            127
          MODULΔγ                         130
          ATTENΔγ                         145
          ~~CON~~ CON                     151
          SAVΔγ | 0→39., X1               152
          SAVΔγ | 40.→1039., Y1           166
     DΔ→  ~~D0~~                          ~~2~~02
      :
      :
```

FRE OGIX 030

```
200000|000000 200204|                P
                     ΔORIG|
200000|301260 200242| STARTΔ→        RFD 60 ΔBEGIN
   001|301712 600335| SSΔY→          SKN 4·10 *CONΔ2
   002|430601 600337|                ³JPX CON*CONΔ4
   003|001600 377604|                DPX  A
   004|022400 200350|                ²LDA  SNIP
   005|007700 200024|                SUB { FULLSCOPΔY}
   006|004700 200002|                JNA  SSΔY+1
   007|031702 200041|                MKN 4·2{SCOPΔ}+13
200010|001260 200012|                REX 60#+2
   011|001660 600051|                DPX 60|CON 1
   012|301712 600335|                SKN 4·10 *CONΔ2
   013|140500 200020|                JPQ  NOSYNCΔY
   014|002400 377610|                LDA  E
   015|007700 600333|                SUB *CONΔ0
   016|004600 200002|                JPA  SSΔY+1
   017|101600 200350|                ¹⁰DPX  SNIP
200020|001260 200001| NOSYNCΔY→      REX 60 SSΔY
   021|001660 600051|                DPX 60|CON 1
   022|021702 200041|                MKZ 4·2{SCOPΔ}+13
   023|430601 600337|                ³JPX CON*CONΔ4

                     ⌐⌐RC
200024|000000 773777|                 FULLSCOPΔY
   025|000000 000000|                 0
                                     SCOPΔ
   026|002400 600333|                   LDA *CONΔ0
   027|201712 600335|                   SKZ 4·10 *CONΔ2
200030|000500 200043|                   JMP #+13
   031|117600 377610|                   ¹¹MUL  E
   032|007200 377610|                   SAB  E
   033|366700 377610|                   ³⁶ADD  E
   034|352400 377610|                   ³⁵LDA  E
   035|306700 200025|                   ³⁰ADD { 0}
   036|227000 200350|                   ²²SCA  SNIP
   037|216700 377610|                   ²¹ADD  E
200040|213400 200350|                   ²¹STA  SNIP
   041|405700 377604|                   ʰTSD  A
   042|430601 600337|                   ³JPX CON*CONΔ4
   043|352400 377610|                   ³⁵LDA  E
   044|306700 200025|                   ³⁰ADD { 0}
   045|227000 200350|                   ²²SCA  SNIP
   046|216700 377610|                   ²¹ADD  E
```

** Reset, Laire Flag + dismiss (seg 0) - go to ΔBEGIN
in sequence 60.

200041

200 026
13
41

** SCOPE| in, TOG, gives in, {SCOPΔ}, TOG

4321

** CONFIG 36 → 32·14
43·21
4·3·2·1
21·43
43·21

** X coord : 4.9-3.9 - (left centered)
Y coord : 2.9-1.9 - signed 1's complement nos

```
200050|430601 600337|        3JPX CON*CONΔ4
   051|000001 000001|     CON 1
   052|000000 000001|      1
   053|000001 000004|     CON 4
   054|000001 000005|     CON 5
   055|604353 330333|      604353330333          * CONFIGURATIONS
   056|734000 000000|     -43,
   057|776000 000000|     -1,
200060|310023 327046|      26848636454•
                          SWGENΔY
   061|001260 200101|        REX 60#+20
   062|001660 600051|        DPX 60|CON 1
 → 063|002401 000003|        LDA CON 3
   064|007600 200060|        MUL { 26848636454•}
   065|140500 200351|        JPQ  COSAΔ
   066|011260 000002|        1REX 60 2
   067|007600 377604|        MUL  A
200070|006700 377604|        ADD  A
   071|007700 200464|        SUB  ALLΔ
   072|410760 200067|        1JNX 60#-3
   073|003401 000003|        STA CON 3
   074|007600 377604|        MUL  A
   075|005600 377604|        COM  A
   076|006700 200464|        ADD  ALLΔ
   077|140500 200377|        JPQ  SQRTAΔ
200100|003401 000004|        STA CON 4
   101|002401 000004|        LDA CON 4
   102|007601 000002|        MUL CON 2
   103|003400 200202|        STA  DΔ
   104|002401 000003|        LDA CON 3
   105|007601 000000|        MUL CON
   106|007700 200202|        SUB  DΔ
   107|005401 000000|        EXA CON
200110|007601 000004|        MUL CON 4
   111|003400 200202|        STA  DΔ
   112|002401 000002|        LDA CON 2
   113|007601 000003|        MUL CON 3
   114|006700 200202|        ADD  DΔ
   115|003401 000002|        STA CON 2
   116|450601 600341|        5JPX CON*CONΔ6
                          LIMITΔY
   117|002401 000003|        LDA CON 3
200120|201711 600335|        SKZ 4•9*CONΔ2
   121|005600 377604|        COM  A
```

```
   123|440601 600340|          4JPX CON*CONΔ5
                               OVADΔY
   124|013000 200126|             1STE #+2
   125|031727 200471|             MKN 1·7 CONTINUE
   126|140500 000000|             JPQ
   127|021000 000000|           17·,
                               MODULΔY
200130|002400 600335|             LDA *CONΔ2
   131|007600 600336|             MUL *CONΔ3
   132|214600 200140|             21JPA #+6
   133|214700 200140|             21JNA #+5
   134|306700 200025|             30ADD { 0}
   135|007200 200127|             SAB { 17·,}
   136|003401 000000|             STA CON
   137|440601 600340|             4JPX CON*CONΔ5
200140|031722 200471|             MKN 1·2 CONTINUE
   141|306700 200025|             30ADD { 0}
   142|007200 200465|             SAB  44Δ
   143|006500 200464|             DSA  ALLΔ
   144|140500 200136|             JPQ #-6
                               ATTENΔY
   145|002400 600335|             LDA *CONΔ2
   146|007601 000003|             MUL CON 3
   147|003401 000000|             STA CON
200150|440601 600340|             4JPX CON*CONΔ5
   151|000001 000000|          CON
                               SAVΔY|0→39·,X1
   152|001506 600151|             ADX 6|CON
   153|001103 200165|             RSX Δ#+12
   154|402000 200476|             LDE  X1
   155|003003 000047|             STE Δ(39·Λ(177777))
   156|410703 200163|             1JNX Δ#+5
   157|001203 200162|             REX Δ#+3
200160|001603 600051|             DPX Δ|CON 1
   161|031721 200471|             MKN 1·1 CONTINUE
   162|420601 600336|             2JPX CON*CONΔ3
   163|001603 200165|             DPX Δ#+2
   164|420601 600336|             2JPX CON*CONΔ3
   165|777777 777730|             (0-39·)V(777777600000)
                               SAVΔY|40·→1039·,Y1
   166|001506 600151|             ADX 6|CON
   167|001103 200201|             RSX Δ#+12
200170|402000 200525|             LDE  Y1
   171|003003 002017|             STE Δ(1039·Λ(177777))
   172|410703 200177|             1JNX #+5
```

FRE OGIX 033

```
   173|001203 200176|               REX  Δ#+3
   174|001603 600051|               DPX  Δ|CON 1
   175|031721 200471|               MKN  1·1 CONTINUE
   176|420601 600336|               ²JPX  CON*CONΔ3
   177|001603 200201|               DPX  Δ#+2
200200|420601 600336|               ²JPX  CON*CONΔ3
   201|777777 776030|               (40·-1039·)V(777777600000)
   202|000000 000000| DΔ→           0
   203|000000 000000| LASQRTΔ→      0
   204|000000 000000| P→            0
   205|000000 000000| SQRTOFΔ→      0
   206|000000 000000| TSΔY→         0
   207|000000 000000| XΔY→          0
200210|000000 000000| in7→          0
   211|000001 237054|               343596·

200212|002400 200241| DSALΔY→     LDA  ALΔ2
   213|013400 200223|             ¹STA  RCLΔY
   214|003401 000003|             STA  CON 3
   215|006701 000004|             ADD  CON 4
   216|007700 200052|             SUB  { 1}
   217|003400 200241|             STA  ALΔ2
200220|001160 377610|             RSX  60 E
   221|001600 377604|             DPX  A
   222|140500 200225|             JPQ  #+3
   223|061260 000000| RCLΔY→      SXL  60
   224|140500 200227|             JPQ  #+3
   225|003460 000000|             STA  60
   226|410660 200223|             ¹JPX  60 RCLΔY
   227|001260 200231|             REX  60#+2
200230|001660 600051|             DPX  60|CON 1
   231|001160 600053|             RSX  60|CON 4
   232|410760 200234|             ¹JNX  60#+2
   233|001160 600054|             RSX  60|CON 5
   234|001660 600053|             DPX  60|CON 4
   235|002400 600335|             LDA  *CONΔ2
   236|005460 600336|             EXA  60*CONΔ3
   237|003401 000000|             STA  CON
200240|460601 600342|             ⁶JPX  CON*CONΔ7
   241|000000 215777| ALΔ2→        DELAY BLOCK
   242|000460 030200| ΔBEGIN→     IOS  60 30200
   243|001206 000001|             REX  6 1
   244|001600 200470|             DPX  TIME
   245|342200 200055| SCCONFΔ→    ³⁴SPG { |604|353|330|333}
   246|011203 000061|             ¹REX
```

Store first usable address ~~for~~ Delay in L(RCLΔY) and set new address in ALΔ2.

** DELAY |in, n is defined by

0
DSALΔY
in
0
-n
1-n

normal entry for delay after setup

** Lo intensity, Left center origin Main Scope

331

** Set TIME = 0

RSX OMINUS {-0}