Massachusetts Institute of Technology

COMPUTATION BOOK

NAME

Number

Course .....

Used from ..... 19 ..., to ..... 19 ...

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

COMPUTATION BOOK

GENERAL INSTRUCTIONS

In all work in which *accuracy* and *ease of reference* are important, much depends upon carrying out the computation in a systematic manner. The following instructions, taken from the *Engineering Department Figuring Book of the Allis-Chalmers Co., serve as a guide in this matter.*

"All computations, of whatever kind, are to be made in these books, except in cases where special blanks may be provided for specific kinds of computation. Computations may be made in ink or pencil, whichever may be more convenient. Pencil figuring should be done with a soft pencil. All the work of computation should be done in these books, including all detail figuring."

"Each subject should begin on a new page, no matter how much space may be left on the previous page. The subject, with the date of beginning it, should be plainly written at the top of the first page of the subject."

"Work should be done systematically, and as neatly as consistent with rapidity. The books are, however, intended for convenience, and no unnecessary work should be done for sake of appearance only. Errors should be crossed off instead of erased, except where the latter will facilitate the work. Work should not be crowded. Paper costs less than the time which would be expended in attempting to economize space in making erasures."

"Where curves drawn on section paper (or sketches) are necessary parts of a computation, they should be pasted in the book, except where specifically otherwise provided for."

"Computations should be indexed, in the back of the book, by the person using the book."

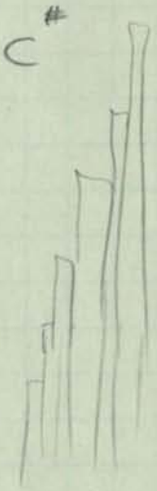
* * * * * * * *"

TECHNOLOGY STORE
HARVARD COOPERATIVE SOCIETY, Inc.
40 Massachusetts Ave., Cambridge 39, Massachusetts

$1.45

semitones	interval
0	unison
1	
2	
3	m 3
4	M 3
5	P 4
6	
7	P 5
8	M 6
9	m 6
10	
11	
12	octave

B
#
A
#
→ G
#
→ F
E
#
D
#
C



Frequency Appendix

X

Master Frequency Chart OCTAL-DECIMAL

Tone control Calculation methods

I.M. time values

Time & tone value equivalence - MUS-HARM

Compiler characters

Proposed Compiler Language

Scan 1 variables

Scan 1 Flowcht

Scan 2 variables

Scan 2 flowcht

} Common Macros

FIO-DEC

} Scan 1 coding

} Scan 2 coding

TE, PC coding

Read flow chart

Read coding (except RT)

X, Complete

Variables, Tables

ERROR flow-chart

Error coding

} MACRO FL→FIO translator

50 FL inverted
51 (translator tables)
52 } RT routines (machine lang. coding)
53
54
55 RND, WA, PPP
56 em
57 PS
58 } up, pf: OUT
59 }
60 key
61 rest / copy
62 pad: OUT
63 up, pf: OUT
64
65
66
67
68
69

B G R

Jan	Feb	Mar
Apr	May	Jun
Jul	Aug	Sep
Oct	Nov	Dec

1	X
2	Z
3	G
4	Y
5	W

Dec. 1, 1961

APPENDIX I

FREQUENCY TABLE

Cycles per Second

C	C♯	D	D♯
16·351	17·323	18·354	19·445
32·703	34·647	36·708	38·890
65·406	69·295	73·416	77·781
130·812	138·591	146·832	155·563
261·625	277·183	293·664	311·126
523·251	554·365	587·329	622·253
1046·502	1108·730	1174·659	1244·507
2093·004	2217·460	2349·318	2489·014
4186·008	4434·920	4698·636	4978·028
8372·016	8869·840	9397·272	9956·056
16744·032			

E	F	F♯	G
20·601	21·826	23·124	24·499
41·203	43·653	46·249	48·999
82·406	87·307	92·498	97·998
164·813	174·614	184·997	195·997
329·627	349·228	369·994	391·995
659·255	698·456	739·988	783·991
1318·510	1396·912	1479·976	1567·982
2637·020	2793·824	2959·952	3135·964
5274·040	5587·648	5919·904	6270·928
10548·080	11175·296	11839·808	12541·856

G♯	A	A♯	B
25·956	27·500	29·135	30·867
51·913	55·000	58·270	61·735
103·826	110·000	116·540	123·470
207·652	220·000	233·081	246·941
415·304	440·000	466·163	493·883
830·609	880·000	932·327	987·766
1661·218	1760·000	1864·654	1975·532
3322·436	3520·000	3729·308	3951·064
6644·872	7040·000	7458·616	7902·128
13289·744	14080·000	14917·232	15804·256

CCCC 16·351 c/s is the lowest note of 32 ft pitch.
CCC 32·703 c/s is the lowest note of 16 ft pitch.
CC 65·406 c/s is the lowest note of 8 ft pitch.
C 261·625 c/s is the so-called middle C of the keyboard.

FREQUENCY CHART

1	2	3	4	5	
40.5477	101.3177	202.6376	405.5000	1013.2004	c 8
32.703	65.406	130.812	261.625	523.251	c 10
42.5132	105.2270	212.4565	425.1351	1052.2727	cs 8
34.647	69.295	138.591	277.182	554.365	cs 10
44.5524	111.3250	222.6520	445.5240	1113.2504	d 8
36.708	73.416	146.832	293.664	587.329	d 10
46.7075	115.6177	233.4402	467.1004	1156.2014	ds 8
38.890	77.781	155.563	311.126	622.253	ds 10
51.1477	122.3177	244.6402	511.5010	1223.2024	e 8
41.203	82.406	164.813	329.627	659.255	e 10
53.5163	127.2351	256.4723	535.1646	1272.3514	f 8
43.653	87.307	174.614	349.228	698.456	f 10
56.1774	134.3770	270.7764	561.7747	1343.7717	fs 8
46.249	92.498	184.997	369.994	739.988	fs 10
60.7774	141.7770	303.7764	607.7754	1417.7733	g 8
48.999	97.998	195.997	391.995	783.991	g 10
63.7234	147.6467	317.5157	637.2335	1476.4676	gs 8
51.913	103.826	207.652	415.304	830.609	gs 10
67.0000	156.0000	334.0000	670.0000	1560.0000	a 8
55.000	110.000	220.000	440.000	880.000	a 10
72.2122	164.4244	351.0514	722.1234	1644.2473	as 8
58.270	116.540	233.081	466.163	932.327	as 10
75.5703	173.3605	366.7416	755.7041	1733.6102	b 8
61.735	123.470	246.941	493.883	987.766	b 10
2026.4010	c6 8				
1046.502	c6 10				
			1.0000		
			1.000		

161.6_{10} MS = $.0001616_{10}$ sec = 0.0000522714533414345_8

Thanks to Expensive Desk Calculator

1 for formula 1, p. 15.

Results by Method 1.

c	1	-	0	0	2	5	5	1		c	3	-	0	1	2	6	4	6		c	5	-	0	5	3	2	2	6
cs	1	-	0	0	2	6	7	4		cs	3	-	0	1	3	3	5	7		cs	5	-	0	5	5	6	7	4
d	1	-	0	0	3	0	2	3		d	3	-	0	1	4	1	1	4		d	5	-	0	6	0	4	6	1
ds	1	-	0	0	3	1	5	7		ds	3	-	0	1	4	6	7	6		ds	5	-	0	6	3	3	7	7
e	1	-	0	0	3	3	2	1		e	3	-	0	1	5	5	0	6		f	5	-	0	6	6	4	3	0
f	1	-	0	0	3	4	7	1		f	3	-	0	1	6	3	4	5		fs	5	-	0	7	1	6	2	4
fs	1	-	0	0	3	6	4	7		fs	3	-	0	1	7	2	3	5		gs	5	-	0	7	5	5	6	4
g	1	-	0	0	4	0	3	4		g	3	-	0	2	0	1	5	7		g	5	-	1	0	3	6	7	4
gs	1	-	0	0	4	2	2	7		gs	3	-	0	2	1	1	3	5		gs	5	-	1	0	4	5	6	3
a	1	-	0	0	4	4	3	2		a	3	-	0	2	2	1	5	0		a	5	-	1	1	0	6	3	7
as	1	-	0	0	4	6	4	4		as	3	-	0	2	3	2	2	2		as	5	-	1	1	5	1	1	0
b	1	-	0	0	5	0	6	7		b	3	-	0	2	4	3	3	5		b	5	-	1	2	1	5	6	4
c	2	-	0	0	5	3	2	3		c	4	-	0	2	5	5	1	3		c	6	-	1	2	6	4	5	4
cs	2	-	0	0	5	5	7	0		cs	4	-	0	2	6	7	3	6										
d	2	-	0	0	6	0	4	6		d	4	-	0	3	0	2	3	0										
ds	2	-	0	0	6	3	3	7		ds	4	-	0	3	1	5	7	4										
e	2	-	0	0	6	6	4	3		e	4	-	0	3	3	2	1	4										
f	2	-	0	0	7	1	6	3		f	4	-	0	3	4	7	1	2										
fs	2	-	0	0	7	5	1	6		fs	4	-	0	3	6	4	7	4										
g	2	-	0	1	0	0	6	7		g	4	-	0	4	0	3	3	6										
gs	2	-	0	1	0	4	5	6		gs	4	-	0	4	2	2	7	1										
a	2	-	0	1	1	0	6	4		a	4	-	0	4	4	3	1	7										
as	2	-	0	1	1	5	1	1		as	4	-	0	4	6	4	4	4										
b	2	-	0	1	2	1	5	6		b	4	-	0	5	0	6	7	2										

lops: 0 0 0 0 5 2

2	~~2999~~ ~~3011~~ ~~3003~~ ~~3002~~	33	141.1	64	598.0
3	2834120 2832530	34	149.5	65	633.6
4	2675415 2673520	35	158.4	66	671.4
5	39.6	36	167.8	67	711.3
6	41.9	37	177.8	70	753.6
7	44.4	40	189.4	71	798.4
10	47.1	41	199.6	72	845.8
11	49.9	42	211.5	73	896.1
12	52.9	43	224.0	74	949.4
13	56.0	44	237.3	75	1005.8
14	59.3	45	251.5	76	1065.7
15	62.9	46	266.4	77	.9905
16	66.7	47	282.3		
17	70.6	50	299.0		
20	74.8	51	316.8		
21	79.2	52	335.7		
22	83.9	53	355.6		
23	88.9	54	376.8		
24	94.2	55	399.2		
25	99.8	56	422.9		
26	105.7	57	448.0		
27	112.0	60	474.7		
30	118.7	61	502.9		
31	125.7	62	532.8		
32	133.3	63	564.5		

```
tone constant test

la,         lat
            add ct
            dac ct
            spa i 40
            clf 4
            spa 40
            stf 4
            law i 7
            dac y
            isp y
            jmp .-1
            jmp la

            variables

start la
```

TONE CONSTANT CALCULATION

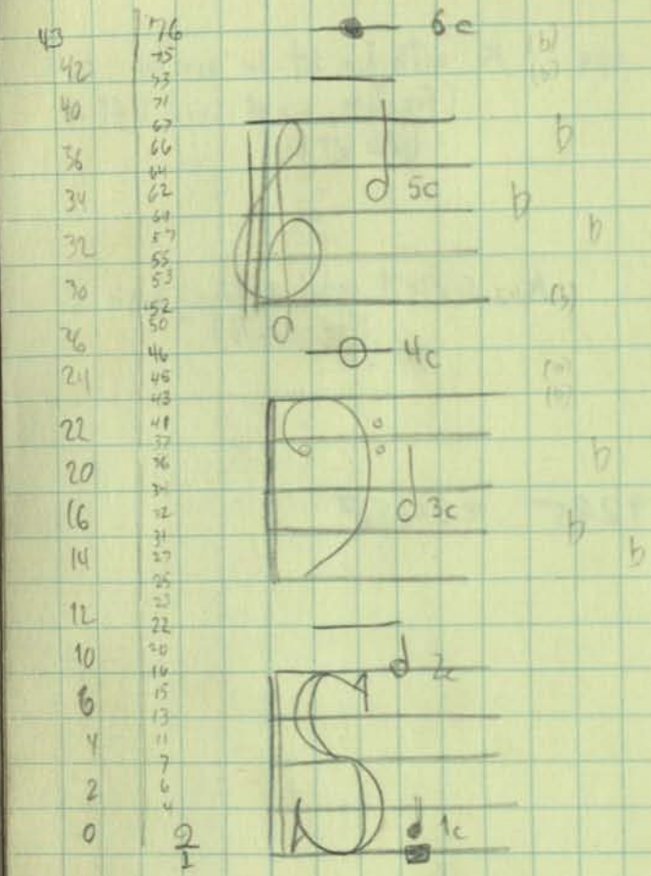
Method 1: $k = fl$ (octal calculations) — l in sec, f in cps, k with h.c. bit in position tested. (fraction, binpt just left of tested bit)

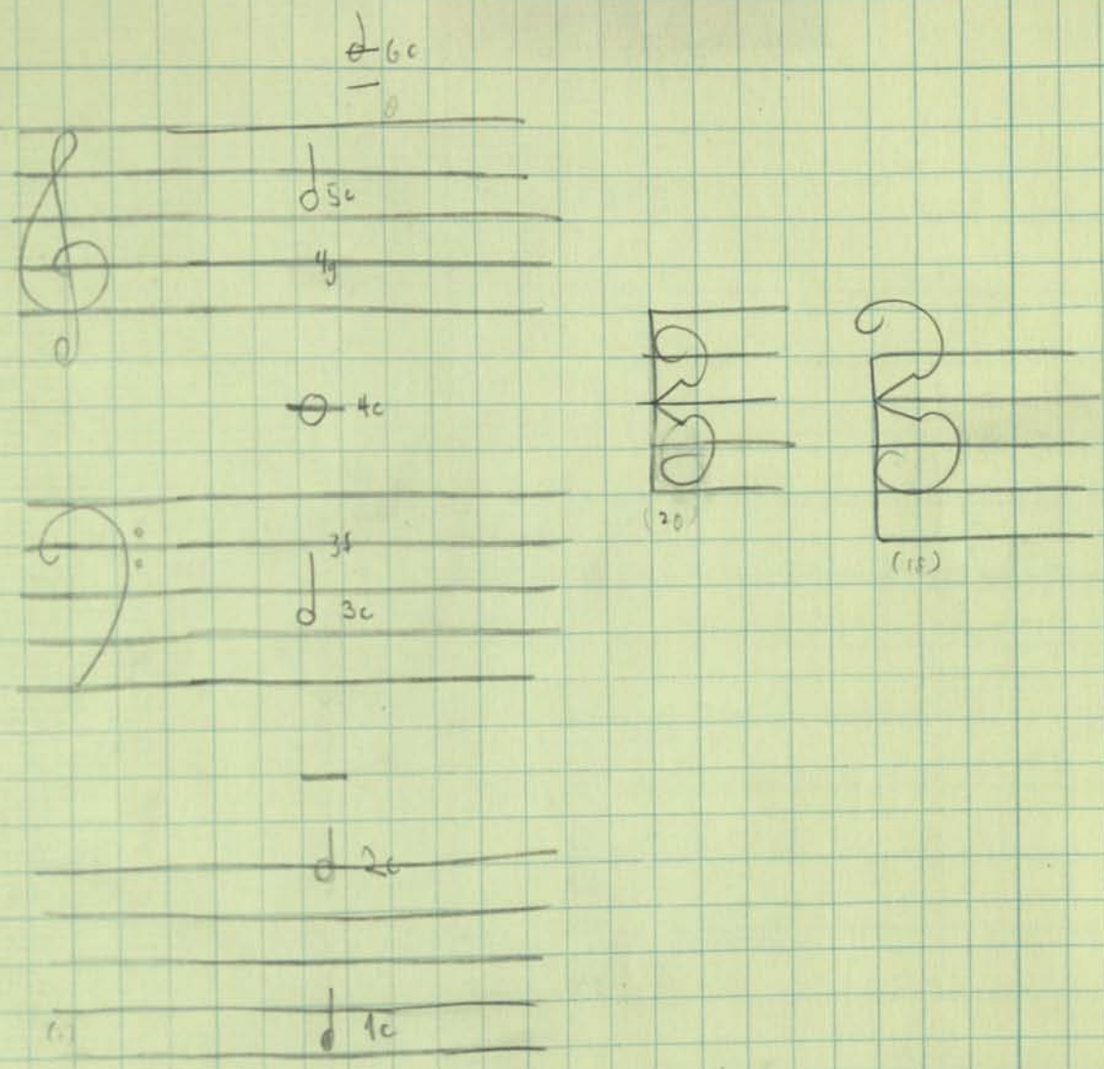
Method 2: $k = fl \times 2^{18}$ (decimal calculations) — l in sec, f in cps, k 6 digit octal integer, high bit tested

$\sqrt[12]{2} = 1.05946\ 30943\ 59295\ 26458$

good fls 125252

121510
114632
111111
105141
100000
074750
070767
066667
063150
060000
055555
052525
050672
046370


43
42
40
38
34
32
30
26
24
22
20
16
14
12
10
6
4
2
0
6c
5c
4c
3c
2c
1c


6c
5c
4g
4c
3f
3c
2c
1c
(20)
(15)

TIME VALUES - INTERPRET MUSIC

		STRAIGHT NOTE	STRAIGHT REST		STAC. NOTE	STAC. REST		LEG. NOTE
WH.	160	170K	10K	20	30K	150K		200K
1/2	70	74K	4K	10	20K	60K	64	100K
1/4	34	36K	2K	4	14K	24K	30	40K
1/8	16	16K	2K	2	6K	12K	14	20K
1/16	7	6K	2K	1	4K	4K	6	10K
1/32	3.4	4K	0	.4	2K	2K		4K

DOTTED - MULTIPLY BY 3/2.

TRIPLET

WH.	116K	7K	20K	105K	125K
1/2	47K	5K	14K	36K	52K
1/4	23K	2K	10K	15K	25K
1/8	11K	2K	6K	5K	13K
1/16	3K	2K	2K	3K	5K
1/32	3K	0	2K	1K	3K

Tone Location Assignment, Time Equivalence

MUS.	HARM.	VAL.
	0	jmp m
20	1	r
35	2	1c
⋮	⋮	⋮
51	16	2c
⋮	⋮	⋮
65	32	3c
⋮	⋮	⋮
101	46	4c
112	57	4a
115	62	5c
⋮	⋮	⋮
131	76	6c
	77	1cps.

Mus Tone → Harm Tone

$20 \rightarrow 1$

$35 \leq n \leq 131 \rightarrow n-33$

else error.

n.7k-0 address for EDP code

Time = $2n-40$, $2n-37$

Tone

MUS.	HARM.	VAL.	in.	tm-mus.
376000	-4000	64		2000
374000	-10000	32	1	4000
370000	-20000	16	2	10000
360000	-40000	8	4	20000
340000	-100000	4	10	40000
300000	-200000	2	20	100000
200000	-377777 (400000)	1	40	200000

MUS time → HARM Time

$-2(\text{tm-mus}) = \text{harm}$

↑ set minus

write word code	F L		FIO DEC	Type	meaning
61	a	✓	a	3	staff above
62	b	✓	b	3	staff below
63	c	✓	c	3	triplet
64	d	✓	d	3	⊥
65	e	✓	e	2	non-staccato, non-legato
66	f		f		
67	g	✓	g	R	grace-note
70	h		h		
71	i		i		
41	j		j		
42	k		k		
43	l	✓	l	2	legato
44	m	✓	m	3	∿∿∿
45	n	✓	n	3	tr
46	o		o		
47	p	✓	p	3	∿∿
50	q		q		
51	r	✓	r	L	rest
22	s	✓	s	2	staccato
23	t	✓	t	3	(dummy)
24	u	✓	u	3	∾
25	v		v		
26	w	✓	w	3	∿∿
27	x	✓	x	3	advance fraction
30	y		y		
31	z		z		
54	-	✓	-	3	♭
57	+	✓	(	3	♯
55	=	✓	)	3	♮
73	.	✓	.	3	dotted
21	!	✓	/	1	bar
33	,	✓	,	R (can be 2 dots)	repeat
00	→	✓	→	1	
36	↑	✓	↑	1	
77	↓	✓	↓	1	
20	0	✓	0		
	:		:		
11	9		9		

Types: 1: terminator
2: accidental modifier
3: note modifier

L: left or right

Dec 9, 1961

Compiler language

Word separators: → → 2 / ≈ → → 2 1

Words: pseudos: no digits, no comma, or pseudo list

numbers: digits, nothing else.

notes: at least one digit or comma; other material if digit.

Scan word: count number of numeric fields, and save values of 1st two.
get fraction value of smallest .
count occurrences of r, g, comma
count characters other than digits
If so characters — go to terminate logic.
If 0 numeric fields and 0 commas, scan against pseudo list.

If 0 non-numeric characters, is number.

Try note: Get XØNEF(r), XØNEF(g), XØNEF(,). [Complaints possible.]

if g∧, error
if ((r + (,) + n) >2), error
if ~;∧((r+g+n)<2), error

set ns for ncr. to save sle.
get init. + final . val. [Time + fract. complaints?]

Scan again:
count a, b, ÷, −, =, c, embellishments [Complaints?]
note s, l, e ~ right Time
count up . x
material copied by comma: ., x, g, c [compl: ÷]

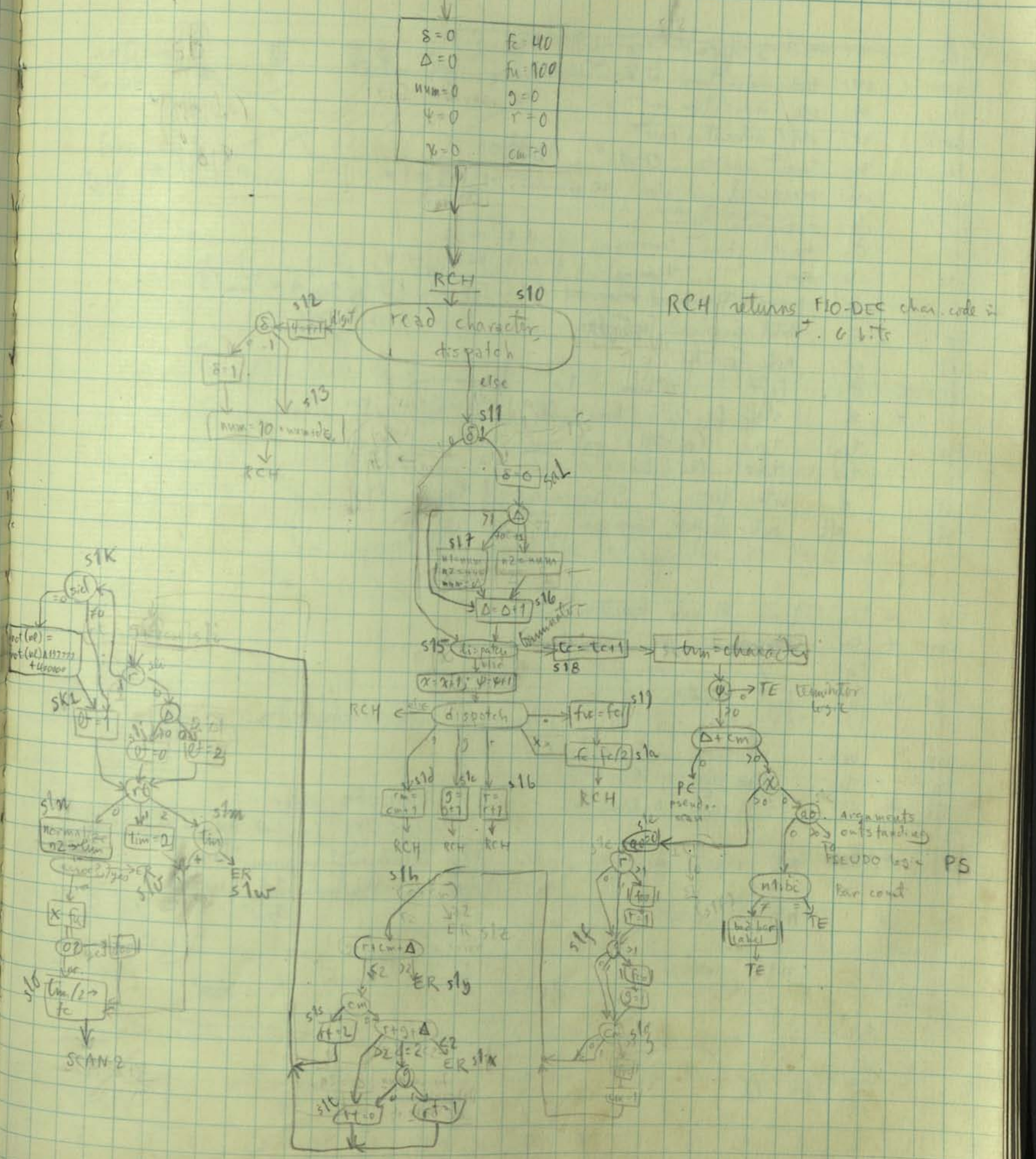
Assemble note: time, sle, tune, tuple

INDICATORS etc. [*: not set on Page 23]

bs		
dfl	δ	δ: preceding character numeric? 1: yes 0: no
ncd	Δ	Δ: number of numeric fields read
num	nem	nem: number being assembled
n1	n1	n1: 1st number in word
n2	n2	n2: 2nd " " " [1st if only 1]
psi	ψ	ψ: total number of characters
chi	χ	χ: " " " non-numeric characters
fc	fc	fc: fraction of whole note assigned to dot
fu	fu	fu: smallest fraction of whole note used by dot
g	g	g: no. of g's
r	r	r: " " r's
cm	cm	cm: " " commas
rt	rt	rt: type of right half: 0: numeric 1: g 2: comma
lt	lt	lt: " " left " 0: numeric 1: r 2: comma
tim	tim	tim: Time (fractional format) not reset so -1 @ beginning of tape
tim	tim	tim terminator (saved for reference).
ac *	ac	ac: number of arguments still desired by pseudo.
bc *	bc	bc: number of chars in measure
tc	tc	tc: terminator count in measure.
chr		character
dig		digit

ERROR: ~~tc no, MQ~~
~~tc~~ AC [for red error]
arg 1 error code

SCAN 1.


δ=0
Δ=0
num=0
ψ=0
χ=0
fc=40
fw=100
ρ=0
r=0
cm=0
RCH
s10
read character, dispatch
s12
digit
δ=1
s13
num=10*num+digit
RCH
else
s11
δ=0
s14
Δ
s17
num=num*10
Δ=Δ+1
s16
s15
ti:=pattern
terminator
fc=fc+1
s18
trm=character
χ=χ+1; ψ=ψ+1
RCH
dispatch
fw=fc
s19
fc=fc/2
s1a
RCH
s1d
s1e
s1f
RCH
RCH
RCH
s1K
s1L
s1M
s1m
ER s1w
s1h
r1=cm+Δ
ER s1y
s1s
ER s1x
s1k
s1j
s1g
ψ
TE
Δ+cm
PC pseudo-op
X
Arguments on stack
PSEUDO logic
n1:bc
Bar count
TE
TE
RCH returns FIO-DEC char. code in fc. 6 bits

SCAN 2

tmp

ld — δ: preceding character numeric? 1 yes 0 no

sv — sv: sle indication of right time } o:e
ss — ss: sle indication rour
sr — sr: staff relocation comm
3i — 3i: triplet indicator 0 nd 3 100000: ?
aci — aci: accidental indicator +0 natural; +1, ±; -1 =
acc — acc: accidental comm
∂ — ∂: embellishment temporary } d:1 +:2
ete — ete: " terminal } n:3 u:4 w:5
fc — fc: previous fractional value of dot. Initially set by Scan 1 = Time/2
fn — fn: total accrued fraction
nf — nf: note forming
lf * lf: left: 0 norm. 1: 2 comma
* st: staff loc (treble, bass, treble) 12, 24, 36
ton — ton: tone (on staff) not used
tne — tne: tone (in title)
nl — nl: note loc set @ beginning of tuple
nft — nft: note formed, time pass

tim * tim Time of note, fractional
cm * cm: comma? 1 yes 0 no
g * g g 1 yes 0 no
n! * n! left member

mm — mm measure units used x3
tu — tu time used by note (x?)
st * st staff comm 0, treble, bass

Tokens: mt measure time

kt key tone

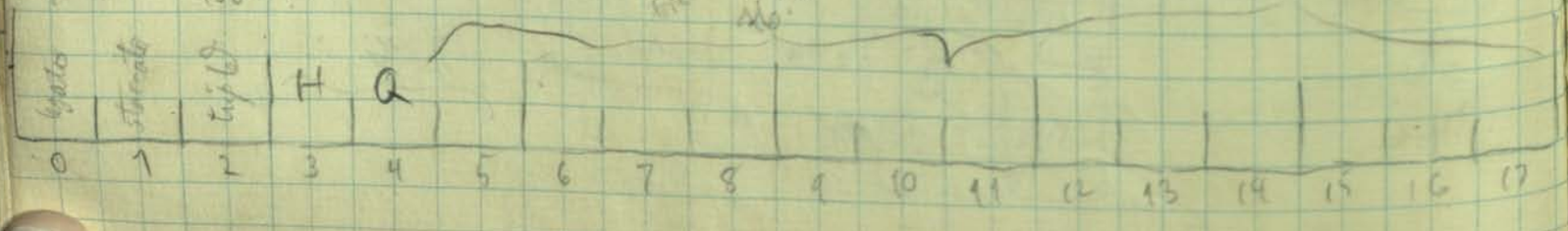
nt natural tone

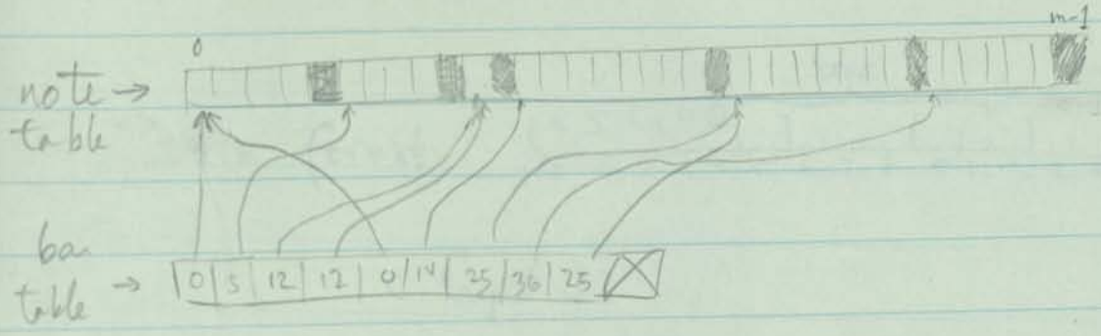
chr: character

700000: error
600xxx: Ger cmd: xx: no. of dots
elf embellishment minimum time
elo " 1st dotted
ele " 2nd "

n: special

Note:





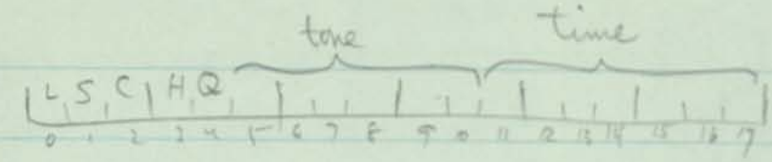
In the note table, each bar is ended by a "note" =600000, shown as a dark square above.

The bar table is a list of entries — 1 per measure — pointing into the note table, to the beginning of the measure. Notice that "copy" is indicated by copying entries from the bar table; no new notes are coded. The square ☒ marking the end of the bar table is =600000.

INTERMEDIATE TAPE FORMAT (output from Phase 1)

400 blank lines
1 wd. m = no. of notes
m wds. notes (including bar end marks)
1 wd. ± checksum of notes
6 blank lines
1 wd n = no. of bars
n wds bar entries
1 wd. ± checksum of bar entries
300 blank lines

NOTE format:



C triplet if 1

Q: "q" articulation

H: "h" articulation

LS: 00 (no effect)

01 "s" articulation

10 "l" articulation

11 end-of-bar marker, ignore rest of word

tone: 0 unused

1 rest

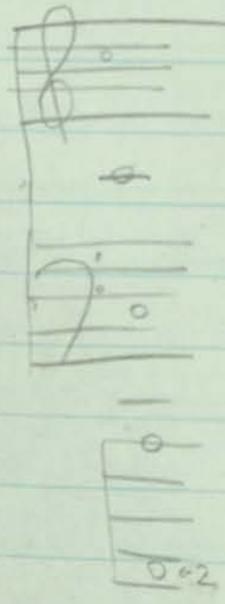
2 – 76 tones →

77 lops. test

time: 1 $\frac{1}{64}$ note

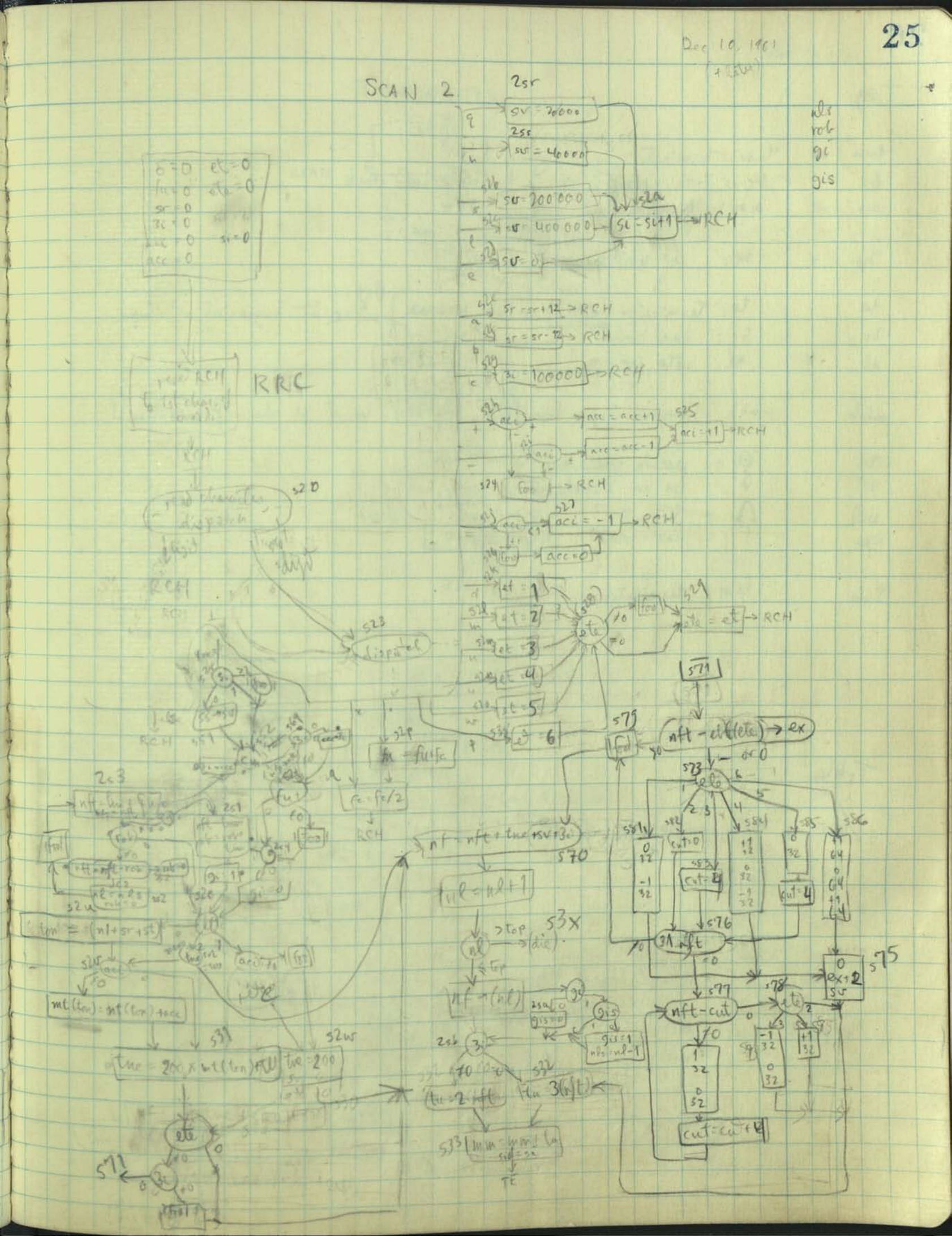
100 whole note

177 o.....



Dec 10, 1961

SCAN 2



tag

tm	*	tm: terminator
bc		bc: bar count, used by start of tags
mm	*	mm: measure unit used (x3)
3u	*	3u: 3x units.
tc		tc: terminator count within measure } set =0 at beginning of tag
bc		bc: bar count
nl	*	nl: note loc.
mbh		mbh: measure begins here. [info. from RCM loc of 1st char of measure]
ptf		θ: pseudo under inv
ze		∫: loc in pseudolist
vd		Δ: terminator? 1: yes 0 no
bpm		⌐: viewer found? 1: yes 0: no
tm		tm: terminator

Tables

bar: testing marking loc. of 1st note of bar

kt: key time

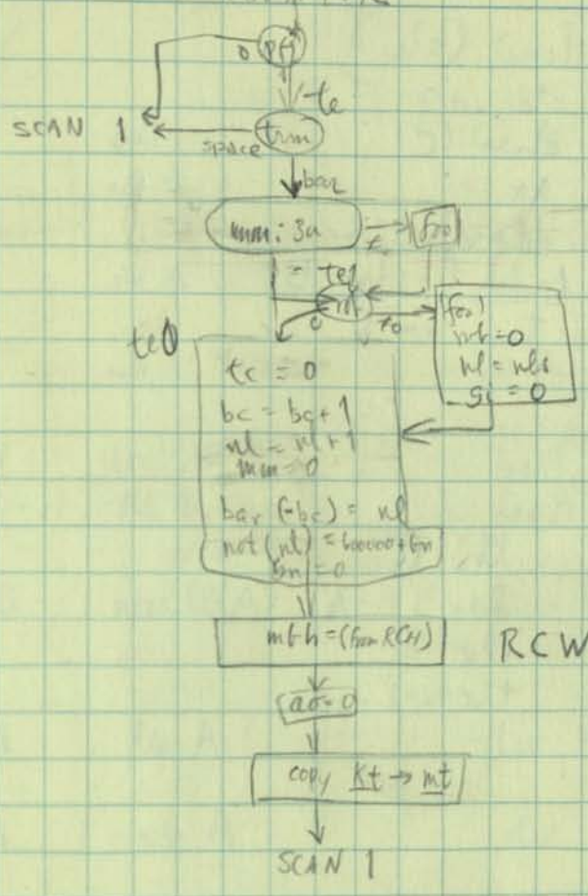
mt: measure time

pnm: pseudo name address table

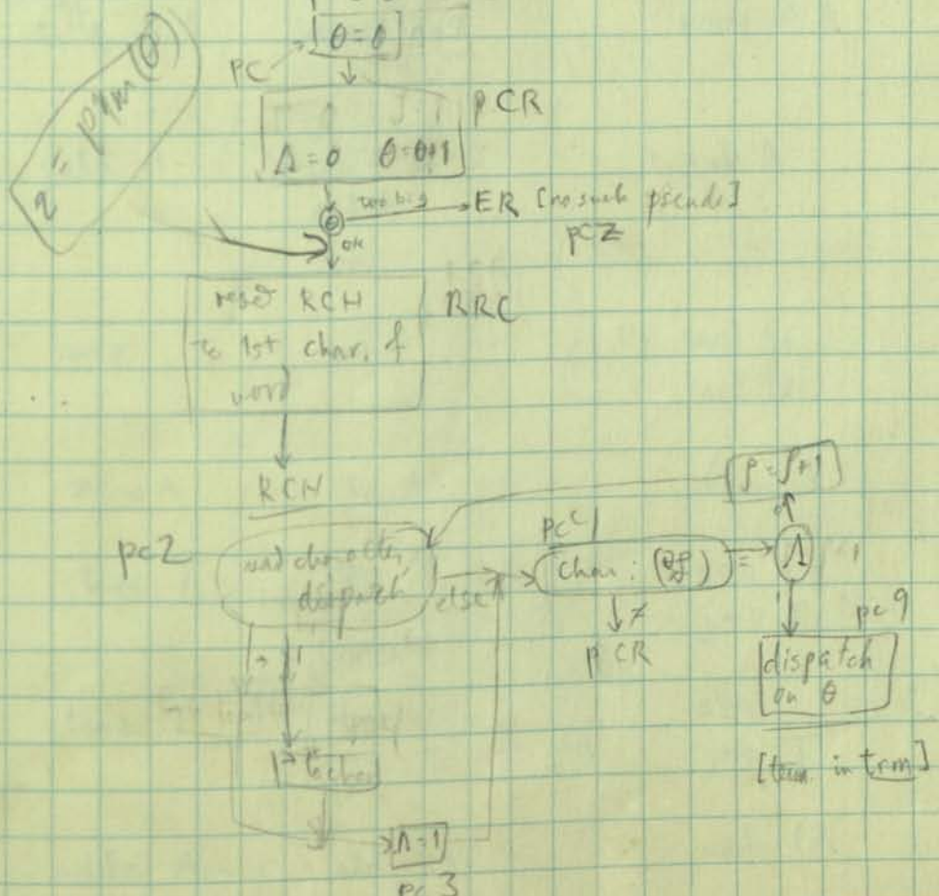
pcd. pc depoten by 0

Dec 11, 1981

TERMINATOR



PSEUDO SCAN



```
clear A, B, C     lda (src A
                  sta .+2
                  ldx (C
                  xx
                  adc .-1
                  add (-src-B
                  tax .-3

                  ldx (C
                  ldx (B-A
                  stx A
                  tix .-1

                  lda A
                  dep .+2
                  lda (C
                  stx
                  inx .-1
                  sas (dest+1
                  jmp .-3
```

Dec. 13, 96

MACROS

TX-0A

load A	cla add A	lac A	cla A
store A	sto A	dac A	sto A
addi A	add A	add A	add A
move A,B	load A sto B	lac A dac B	cla A sto B
set A,B	move (B,A	move (B,A	
loadq A	ura A	lio A	
storeq A	szb A	dio A	
moveq A,B	ura A szb B	lio A dio B	
setq A,B	moveq (B,A	moveq (B,A	—
→ sult A	com add A com	sub A	
swap	iol	rcl 9s rcl 9s	
zero A	stz A	dzm A	
tze T	tze T	szf i jmp T	
tnz T	tze +2 ta T	szf jmp T	
addc A	add (A	add (A	

tspl T	tsn ,+2 tha T	-sma jmp T	
tsmi T	tsn T	spa jmp T	
tsel A, T	com add A tze T	sad A jmp T	
loada A	cla add CA	law A	lda CA
tsmq T	iol tsn T	spc jmp T	tpl +2 tra T
atsmq	iol iol	ϕ	
tspq T	iol 40 tsn T	spc i jmp T	tpl T
atspq	iol 40 com iol	ϕ	
ftsmq T	tsmq T iol	tsmq T	
ftspq T	tspq T com iol	tspq T	

atrl A	com add A	∅	
trnl A, T	com add A tra .+2 ba T	sas A jmp T	
ftrnl A, T	trnl A, T atrl A	trnl A, T	
ftrel A, T	trel A, T atrl A	trel A, T	
dispatch U	add (add) U sto .+2 cla xx add (tra sto .+1 xx	add (U dap .+1 jmp i	axr trx 0
execute U	add (add) sto .+2 cla xx sto .+1 xx	add (U dap .+1 xj	axr Ux 0 sh .+1 xx
lookup V	add (add) V sto .+2 cla xx	add (V dap .+1 lac	axr lax V
step J, I	(rad J add I sto J	lac J add I dac J	
step 1 J	ado J	idx J	
stepa J, I	ado J adelc I-1 sto J	law I add J dac J	(ra J add (I sto J

	halve	shr	sar 1s
	x2to1	cyl	rol 1s
	x2to2	cyl cyl	rol 2s
	x2to4	repeat 4, cyl	rol 4s
	x2to7	repeat 7, cyl	rol 7s
value in AC	putback U, Q	add (sto U sto ,+3 cla add Q xx	add (U dap .+2 lac Q dac
	call S	lm (tra .+2 tra S	jda S
	answer X	sln X	0 dap X lac .-2
	exit	xx	jmp
	reached E, T	add (-E tze T	sad (E jmp T
	single0	tra .+5 tze .+5 cla add (1 tra 1.+2 cla	spa cla sza law 1
	single1	tra .+2 tra .+2 cla	spa cla

	xno	tze .+3 cla tha .+2 add (1	sza law i 1 add (1
	goto L	tra L	jmp L
	search W, N, ERR	alo add (add W sto .+2 lcc xx tze .+5 ado .-2 add (-add-W-N+1 tnn .-5 tra ERR add .-6 add(-add-W	dac t1 law W dap .+2 lac t1 sad jmp .+5 idx .-2 sas (sad W+N jmp .-5 jmp ERR lac .-6 add (-sad-W
	tles C, T	cma add C tnn .+2 tra T	sub C spa jmp T
	tlesc C, T	add (-C+1 tnn T	sub (C spa jmp T
	tequ C	xor C ana	and C
AC ≥ C	tgrel C, T	cma add C tnn T	sub C sma jmp T
AC > C, →	tgrec C, T	add (-C tnn .+2 tra T	sub (C sma +sza -skp jmp T
	atgrec C	add (C	add (C

xno
adx
lcc
adx W
tze .+3
tix .-3
tra ERR
xac

Doc. 14, 1961

complement	com	cma
tstd Y, Z	load Y tze Z	lac Y sza i jmp Z
tstl Y, Z	load Y tze .+2 tha Z	lac Y sza jmp Z
x10dec	cyl sto t1 cyl cyl add t1	ral 1s dac t1 ral 2s add t1
tral U, V	com add U tze .+2 tha V	sas U jmp V
clear	cla	cla
complaint [U]	load bc load tc goto er U	load bc load tc goto er U
error [U]	load bc load tc complement goto er U	load bc load tc complement goto er U
halfof V	load V halve store V	load V halve store V
double Q	load Q x2to1 store Q	load Q x2to1 store Q

G?

goto P	(oad P add (ma sta .+1 xx	jmp i P
istpa A, B	ade A add (-B-1 sto A	law i B add A dac A
copy H, I, N	(h (h H sh .+3 (h (sh I sh .+2 xx xx ado .-2 ado .-2 add (- sh-I-N+1 tnn .-5	law H dap .+3 law I dap .+2 lac N dac I idx .-2 idx .-2 sas (dac I+N jmp .-5
✓ tstm Y, Z	(oad Y tnn Z	lac Y spa jmp Z
✓ tstp Y, Z	(oad Y tnn .+2 tra Z	lac Y sma jmp Z
✓ tstel Y, Z, A	(oad Y tnel Z, A	(oad Y tnel Z, A
✓ tstnl Y, Z, A	(oad Y tnel Z, A	(oad Y tnel Z, A
✓ write P	load (tze P ll (add .+2 tne wr	law P jda wr
✓ type Q	load Q lookup xyz pwo	law Q typ
✓ print F	(oad F pwo pwo pwo	lac F repw 3 → rdbs + typ

FIO-DEC

00	→	
01	1	
02	2	
03	3	
04	4	
05	5	
06	6	
07	7	
10	8	
11	9	
12		
13	stop	
14		
15		
16		
17		
20	0	
21	/	?
22	s	
23	t	
24	u	
25	v	
26	w	
27	x	
30	y	
31	z	
32		
33	,	=
34	Б	
35	Я	
36	→	
37		

40	·	—
41	j	
42	k	
43	l	
44	m	
45	n	
46	o	
47	p	
50	q	
51	r	
52		
53		
54	-	+
55	)	]
56	.	|
57	(	[
60		
61	a	
62	b	
63	c	
64	d	
65	e	
66	f	
67	g	
70	h	
71	i	
72	↓	
73	.	×
74	↑	
75	←	
76		
77	↓	

00

	0	4	14	10
0	→	4		8
1	1	5		9
3	3	7		stop
2	2	6		

	0	4	14	10
0	0	u	Б	y
1	/	v	Я	z
3	t	x		,
2	s	w	→	

	0	4	14	10
0				

40

	0	4	14	10
0	·	m	-	q
1	j	n	)	r
3	k	p	(	
2	k	o	-	

00

	0	4	14	10
0		↓	↑	h
1	a	e	←	i
3	c	g	2	.
2	b	f		↓

20

	0	4	14	10
0				

12-14-61

SCAN 1

```
s1,     zero ldl
        zero ucd
        zero num
        zero psc
        zero chr.
        zero g
        zero r
        zero cm
        set fc, 20
        set fe, 40
s10,    call rch
        store chr
        trze s11
        tjuc 20, s11
s12,    step1 psi
        test1 ldl, s13
        set ldl, 1
s13     load chr
        tnel (20, s14
        clear
s14,    store dig
        load num
        x10 dec
        addi dig
        store num
        goto s10

s11,    test0 ldl, s15
        zero ldl
        load ucd
        tjuc 1, s16
        tnze s17
        move num, n2        zero num
        goto s16
s17,    move num, n1
        store n2
s16,    step1 ucd

s15,    load chr
        trze s18
        tnel (21, s18
        step1 chr
        step1 psi
        load chr
        ftnel (73, s19
        ftnel (27, s1a
        ftnel (51, s1b
        ftnel (67, s1c
        tnel (33, s1d
        goto s10

s19,    move fe, fn
s1a,    load fe
        halve
        store fe
        goto s10
s1b,    step1 r
        goto s10
s1c,    step1 g
        goto s10
s1d,    step1 cm
        goto s10

s18,    step1 te
        move chr, tem
        test0 psi, te
        load ucd
        addi cm
        trze pc
        test1 chr, s1e
        test1 ac, pl
        load n1
        tnel bc, te
        complain flexo blk
        goto te
```

```
s1e,    load r
        tbsc 2, s1f
        complaint flexo tmr
        set r, 1
s1f,    load g
        tbsc 2, s1g
        complaint flex. tmg
        set g, 1
s1g,    load cm
        tbsc 2, s1h
        complaint flexo tmc
        set cm, 1

s1h,    load g
        addi cm
        tgtec 1, s1z
        atgtec 1
        addi r
        addi ucd
        tgtec 2, s1y
        test1 cm, s1i
        load r
        addi g
        addi ucd
        tbsc 2, s1x

s1i,    load cm
        x2to1
        addi g
        store rt
        test1 r, s1k
        test1 ucd, s1j
        load (2
        goto s1l
s1j,    clear
        goto s1l
s1k,    load (1
s1l,    store lt
        test0 rt, s1m
        tgtec 1, s1m
        zero tim
        goto s1o

s1m,    load tim
        tmi s1w
        goto s1o
s1n,    set tim, 40
        set t2, 1
        move n2, t1
        trze s1v
        tgtec 40, s1v
s1q,    halfof t1
        trze s1p
        halfof tim
        halfof fu
        double t2
        goto s1q
s1p,    load n2
        tmnl t2, s1v
        test1 fu, s1o

        complaint flexo dtu
s1o,    load tim
        halve
        store fc
        goto s2
```

SCAN 2

```
s2,   zero ldl
      zero fu
      zero sr
      zero 36
      zero acc
      zero acc
      zero et
      zero etc
      call rcr

s20,  call rch
      store chr
      tze s21
      tqnec 20, s21
      testo ldl, s22
      goto s20
s22,  set ldl, 1
      goto s20

s21,  testO ldl, s23
      zero ldl
      testI rt, s23
      move ss, sv
s23,  load chr
      search s2z, 24, s20
      dispatch s2y

s2a,  move rs, sv
      goto s20
s2b,  set ss, 200000
      goto s20
s2c,  set ss, 400000
      goto s20
s2d,  zero ss
      goto s20
s2e,  stepa sr, 14
      goto s20
s2f,  isbpa sr, 14
      goto s20
s2g,  set xi, 100000
      jfw s20

s2h,  load acc
      tmi s24
      stepI acc
      goto s25
s2i,  load acc
      tmi s24
      isstepa acc, 1
s25,  setI acc, 1
      goto s20
s24,  complaint flexo nor
      goto s20
s2j,  load acc
      teq (1, s26
s27,  set acc, -1
      goto s20
s26,  complaint flexo nor
      zero acc
      goto s27
s2k,  load (10000
      goto s28
s2l,  load (20000
      goto s28
s2m,  load (30000
      goto s28
s2n,  load (40000
      goto s28
s2o,  load (50000
s28,  store et
      testO etc, s29
      complaint flexo tme
s29,  move et, etc
      goto s20
s2p,  stepI fu, fc
s2q,  halfof fc
      goto s20
```

s38, load (60000 goto s28

```
s2r,    test1 cm, s2t
        test1 g, s2s
        load tim
        addi fu
        addi 3i          [store nft
        goto s2s
s2t,    load fu
        addi 3i
        trac s2s
        complain flexo tic

s2s,    test0 lt, s2u
        trac 1, s2v
        test0 aci, s2w
        complain flexo air
s2w,    set tne, 100
        goto s30

s2u,    load nl
        addi sr
        addi st
        store ton
s2v,    test0 aci, s31
        load ton
        looking at
        addi acc
        store t1
        load ton
        pullback NT, t1

s31,    load to
        lookup mt
        x2 to 4
        x2 to 2          ] → x2 to 6
        store tne
s30,    load nft
        addi tne
        addi eta
        addi sss
        store nf
```

```
step1 nl
~~trac to, s32~~
pullback a, nf
test0 3i, s32
load tim
addi fu
x2 to 1
store tu
goto s33
s32,  load tim
      addi fu
      store tu
      x2 to 1
      addi tu
      store tu
s33,  step mm, tu
      goto te
```

TERMINATOR

```
t1,     testO tm, s1
        load mm
        tnel 34, te1
        complaint flexo mkw
t1,     zero te
        stop1 te
        complement
        postack bar, nl
        zero mm          zero bn
        call rcw
        store mbh
        copy kt, mt, 75
        goto s1
```

```
        stop1 nl
        load bn
        addi (600000
        store t1
        load nl
        postack 0, t1
```

```
        addi nl
        tgtec all, 53
        load te
```

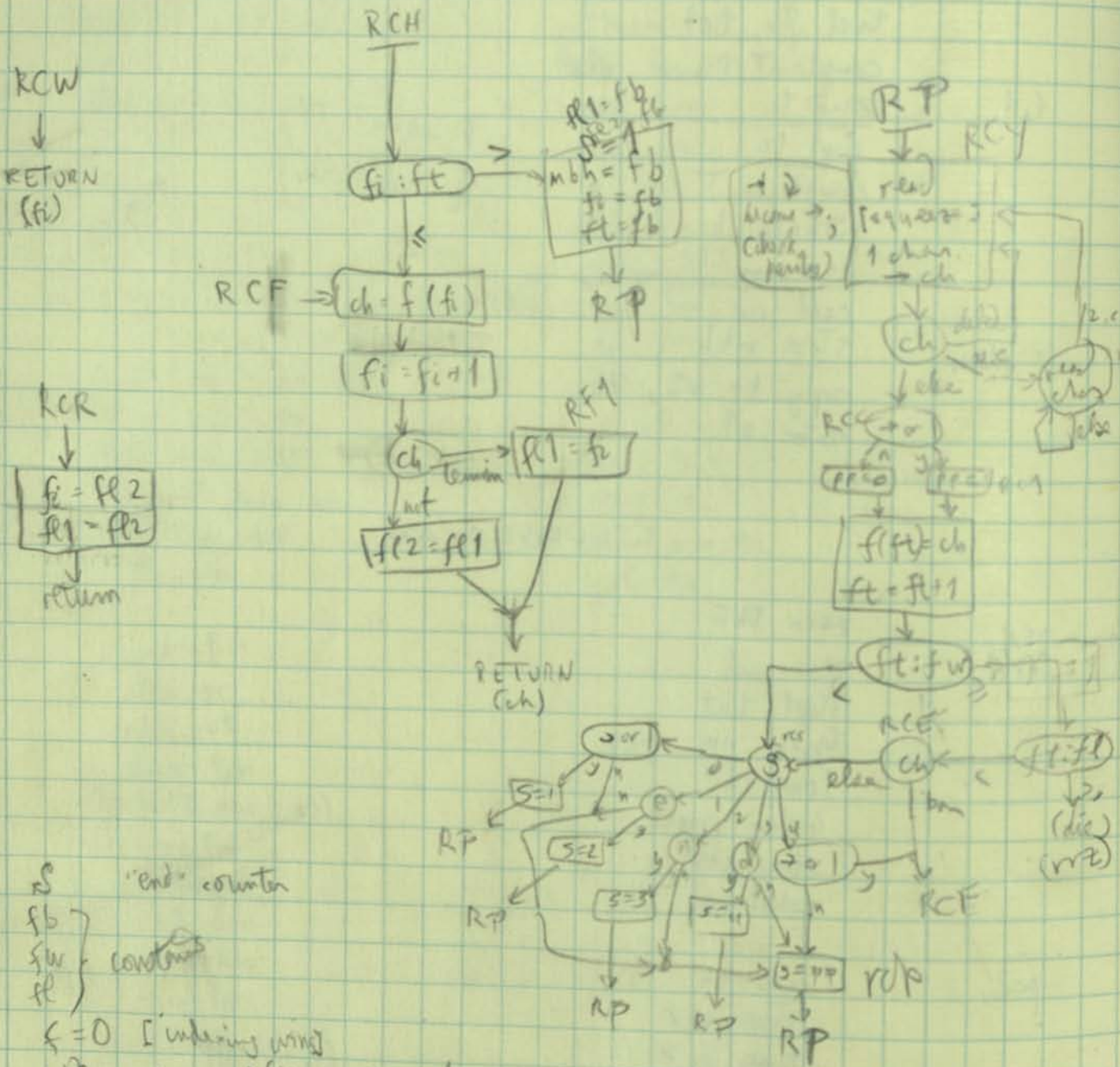
```
        addi te
        tgtec all, 53
```

PSEUDO SCAN

```
pc,     zero tht
pc1,    zero ucl
        stop1 tht
        tgtec npi, pc2
        load tht
        lookup pnm
        store zet
        call rer

pc2,    call rch
        store chr
        tnel pc3
        tnel (21, pc4
        zero chr
pc3,    sett ucl, 1
pc4,    load zet
        lookup 0
        tnel chr, pc1
        test1 ucl, pc9
        stop1 zet
        goto pc2
pc9,    load tht
        dispatch pcd
```

READ CHARACTER



bgs
- S "end" counter
- fb, fw, ft } constants
- f = 0 [indexing word]

inbb
- mbb assume [flexo] begins here
- fi — f index
- ft — f-buf
- ch — character from tape
- pp — saves → fi to
- ft1 } save fi in f of beginning of word
- ft2 }

READ CHARACTER

```
rcw,    answer rwx
        load fi
rwx,    exit

rcr,    answer rrx
        move fl2, fi
        store fl1
rrx,    exit

rch,    answer rcx
        load ft
        tqnel fi, rcf
        set mbh, fb
        store fi
        store ft
*  rcy, call rt
        store ch
        Tunl (74, rcc
*  rcu, call rt
        tunl (72, rcu
        goto rcy

rcc,    load ch
        tuze rc1
        tnel (21, rc1
        clear
        goto rc2
rc1,    load (1
rc2,    store pp
        load ft
        putback f, ch
        stgl ft
        tbasc fw, rcs
        load ft
        tbasc fl, rce
        goto rc2
rce,    load ch
        tnel (21, rcf

rcs,    load bgs
        diswith ch, rdt

rdt,    rd0
        rd1
        rd2
        rd3
        rd4

rd0,    tuze rda
        tunl (21, rdp
rda,    set bgs, 1
        goto rcy
rd1,    tunl (45, rdp
        set bgs, 2
        goto rcy
rd2,    tunl (45, rdp
        set bgs, 3
        goto rcy
rd3,    tunl (64, rdp
        set bgs, 4
        goto rcy
rd4,    tuze rcf
        tnel (21, rcf
rdp,    move pp, bgs
        goto rcy

rcf,    load fi
        lookup f
        store ch
        stgl fi
        load ch
        tuze rf1
        tnel (21, rf1
        move fl1, fl2
        goto rff
rf1,    move fi, fl1

rff,    load ch
rcx,    exit
```

s2z,	33	/ !	s2y,	s2a
	67	/ g		s2a
	22	/ s		s2b
	43	/ l		s2c
	65	/ e		s2d
	61	/ a		s2e
	62	/ b		s2f
	63	/ c		s2g
	57	/ (+		s2h
	54	/ -		s2i
	55	/) =		s2j
	64	/ d		s2k
	44	/ m		s2l
	45	/ n		s2m
	47	/ p		s38
	24	/ u		s2n
	26	/ w		s2o
	73	/ .		s2p
	27	/ x		s2q
	21	/ / \		s2r
	00	/ →		s2r

complaints

- bbl	bad bar label	· uat	unavailable tone
- tmr	too many r's	· aor	attempted out of range
- tmg	too many g's	- tms	too many s, l, e.
- tmc	too many c's	· ei	embellishment in illegal time
- dtu	dot underflow	· eth	embellishment in triplet
- nor	natural overrules	eor	embellishment on 4 notes
- tme	too many embellishments		
- tic	time in comma note		
- aic	accidental in rest		
- mlw	measure length wrong		
* - tmf	too many fields		
* - tff	too few fields		
* - unc	unprepared comma		
* - ert	erroneous tune		
* - nps	no such pseudo		
* - ilr	"rest" illegally placed		
* - ilc	"copy" illegally placed		
* - blc	bad left arg. to copy		
* - brc	bad right arg. to copy		

Variables

Val
ucd
num
n1
n2
psi
chi
Sc
Fu
g
r
cth
rl
ftl
tim
trm
ao
bc
tc
chr
dig
sr
ss
sn
3i
acj
ace
el
eta
nf
ton
tna
nl
nft
mm
tu
3u
mbh
tlf
zet
ucl

Gam
Ugs
fi
ft
pp
ch
fl1
fl2
chy
arg
win
etc
Gle
xjp
tj1
emp
rh
t1
t2
st
rn
_bn
riw
rij
ril
np
pfu
inl
_cbh
pfi

fpi
mpt

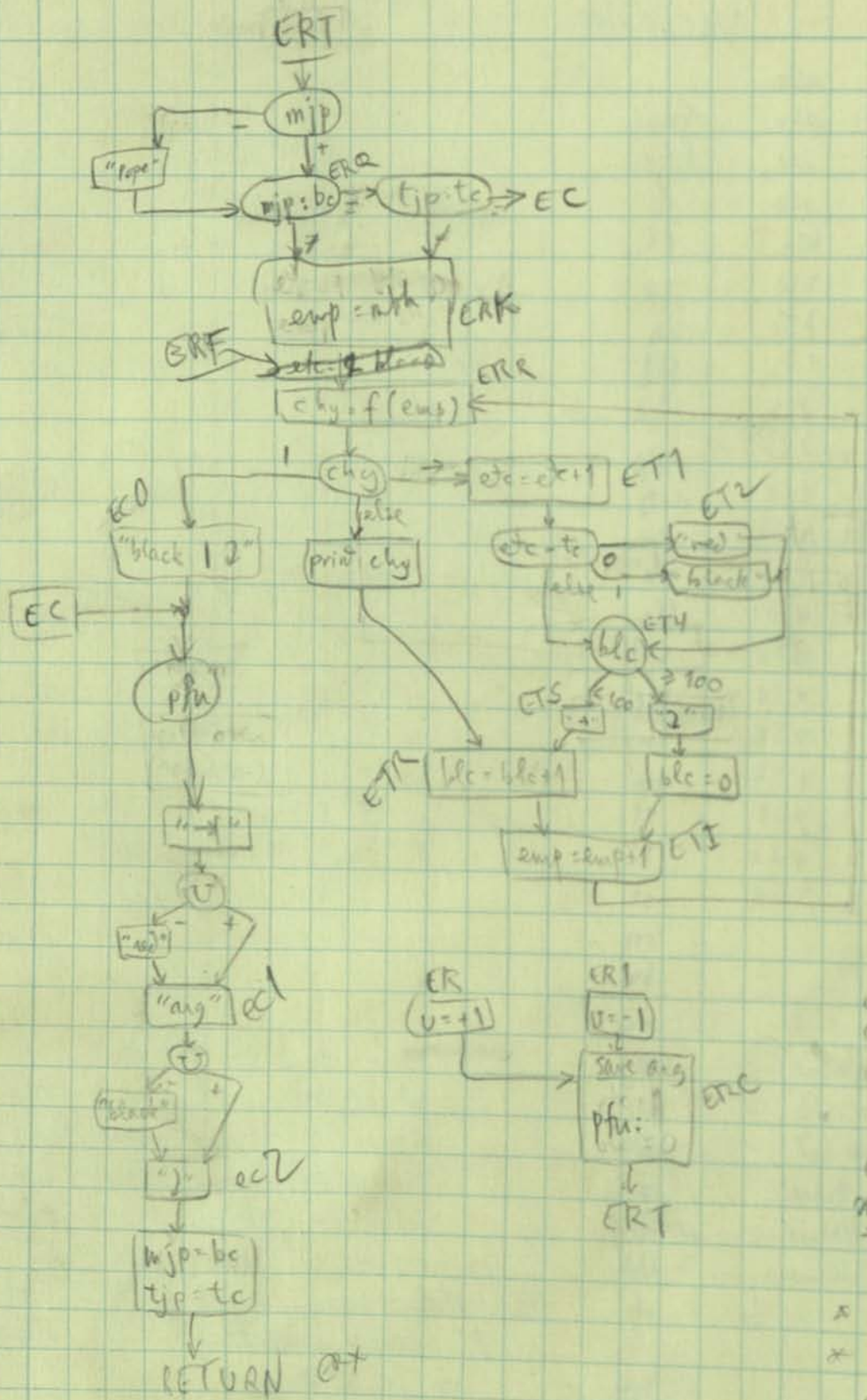
Tables

mt
kt
nt
tau
bnm
ti
pbb

Constants

f
fb
fw
fl
all

ERROR



chg
arg
win
etc
blc
mjp
tjp

* bc
* tc
emp
* mth

rf: +1 black/-1 red

ERROR

```
er,     answer erx
        store arg
        set win, 1
        goto erc
erl,    answer erx
        store arg
        set win, -1

erc,    set dc, 1
        zero blc
eid,    testp wjp, erq
        write erq
        text /DiTto -> err -> is -> human ... to -> forgive, -> divine. dd/
erq,    testwl wjp, bc, ert
        testel fjp, tc, ec
eO,     move wbh, emp

ere,    lookup #
        store chy
        type et1
        thel (21, ec0
        type chy
        goto etr

et1,        step1 etc
            sult tc
            tize et2
            thel (1, et4
             call blk
              goto et4
et2         call red
et4,        load blc
            tlesc ets
            type (77
             zero blc
             goto eti
ets,        type 10
etr,        step1 blc
eti,        step1 emp
            goto err

red, answer rex
        testm rb, rex
        type (35
        set rb, -1
rex,    exit

blk, answer blx
        testp rb, blx
        type (34
        set rb, +1
rex,    exit

ec0,    call blk
        type (21
        type (77

ec,     call rnd
        call opt
        type (36
        testp win, ec1
        call red
ec1,    print arg
        testp win, ec2
        call blk
ec2,    type (77
        move bc, wjp
        move tc, tjp
erx,    exit
```

MACRO Convert 10 : lc
11 : uc

nums → nums
l.c. b's → l.c. b's
u.c. b's → u.c. b's
→, ⊣, ⊤ → →, ⊣, ⊤
$ → ϕ
– → –
+ → +
/ → +
| → /
: → .
; → ,
(,) → (,)
= → =
— → ϕ
u.s. nums, → uc nums,
blank → blank

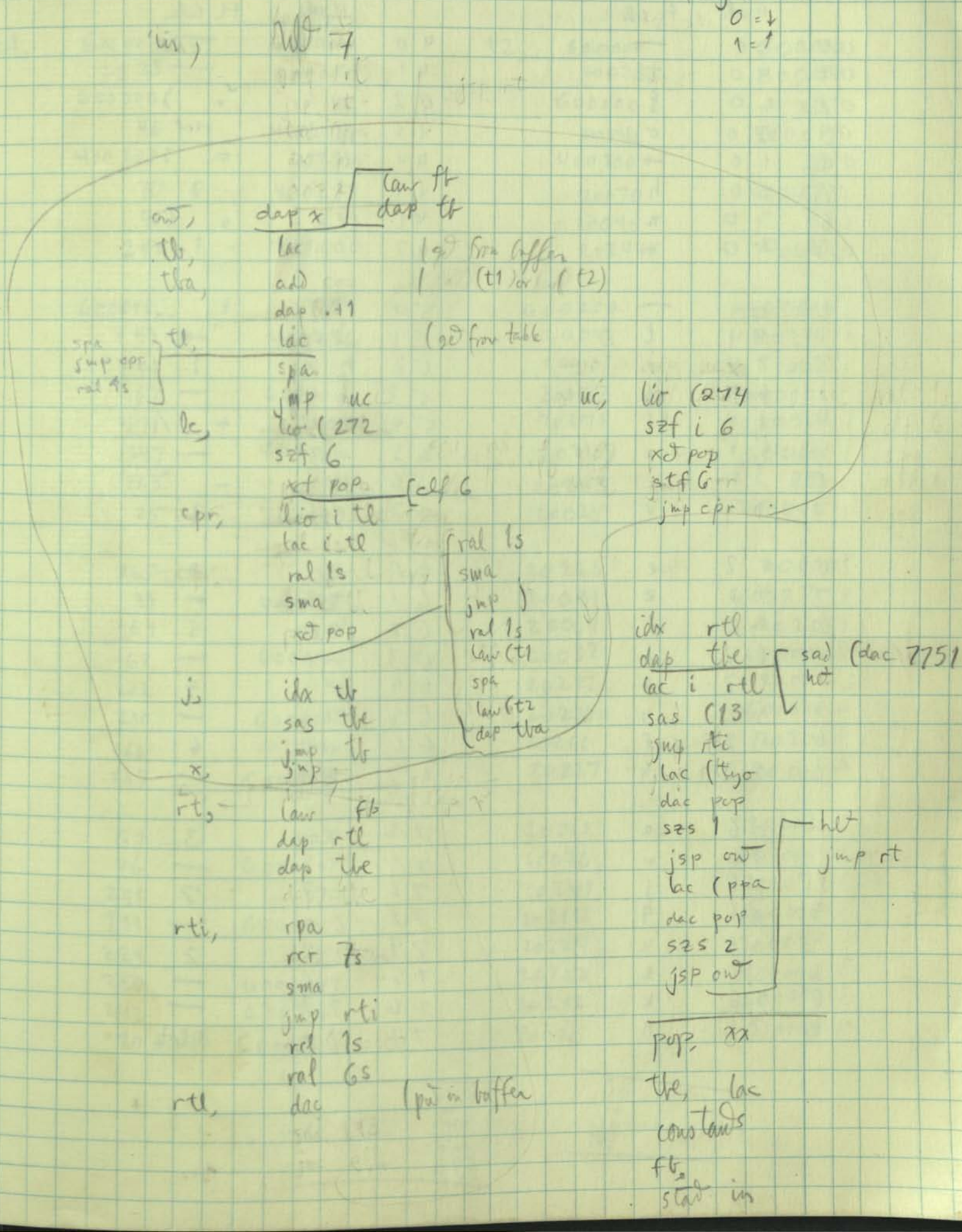
1: non-case	1: non-punch	1: col 12	1: uc

12 case

Punch code

1001

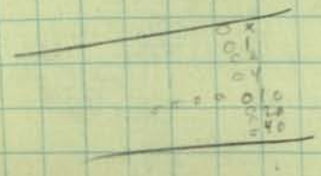
flag 6 : case
0 = ↓
1 = ↑

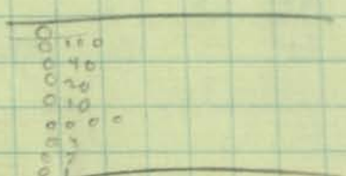


FL code — OCTAL VALUE "inverted" [i.e. as read by PDP]

	Fl.val.	Fl code
00	00	—
01	40	t
02	20	ξ
03	60	σ
04	10	→
05	50	h
06	30	n
07	70	m
10	04	—
11	44	l
12	24	r
13	64	ξ
14	14	i
15	54	p
16	34	c
17	74	v
20	02	e
21	42	z
22	22	d
23	62	b
24	12	s
25	52	y
26	32	f
27	72	x
30	06	a
31	46	w
32	26	j
33	66	9
34	16	u
35	56	q
36	36	k
37	76	0

	Fl.val.	Fl Code
40	01	—
41	41	—
42	21	.)
43	61	STOP
44	11	= :
45	51	2
46	31	, (
47	71	↑
50	05	1 —
51	45	→
52	25	1
53	65	—
54	15	+ /
55	55	—
56	35	-
57	75	↓
60	03	8
61	43	←
62	23	5
63	63	—
64	13	4
65	53	—
66	33	6
67	73	—
70	07	3
71	47	—
72	27	7
73	67	—
74	17	2
75	57	—
76	37	—
77	77	delete





t1,

600000
20023
600000
46
400200
70
45
244

600000
43
51
67
271
247
263
25

265
31
64
62
222
230
266
227

61
26
241
211
224
250
242
20

600000
600000
40
400013
20233
400277
233
740000

221
400236
1
600000
20054
600000
54
700000

10
400075
205
600000
4
600000
206
600000

203
600000
7
600000
2
600000
600006
600006

t2,

600000
20023
600000
20046
400200
20070
20045
20244

600000
20043
20051
20067
20271
20247
20263
20025

20265
20031
20064
20062
20222
20230
20266
20227

20061
20026
20241
20211
20224
20250
20242
20020

600000
600000
255
400013
73
400277
57
740000

6000000
400236
20001
600000
20054
600000
256
700000

20010
400075
20205
600000
20004
600000
20206
600000

20203
600000
20007
600000
20002
600000
600000
600000

Dec. 27 READ TAPE (machine lang.)

PDP

rt, 0
lap rtx
rt2, rpa
rcr 7s
spa
jmp .-3
dio t1
law 1000
add t1
dap .+2
law 2525
670000 (ror
sma
jmp rt2
law 77
and t1
sad (77
cla
sad (36
cla
sad (13
jmp rt2
rtx, jmp

TX-0

rt, shr rtx
rt2, rlc
dlo
add (333330
cry
dlo
add (333330
cry
dlo
add (333360
cry
dlo
add (333000
cry
dlo
add (333000
cry
lw (400037
ana -10 sto t4
dac (add t1
sto .+2
cla
rsx (add rt1+...
trn rtp
rtx, xx | tra (o...

rtp, cyl + 40
trn rt2
cla
add t4
lio
tra rt2
t1
t4

READ TAPE
TX-0 TABLE

rt1,	-0	#
	-0	#
	65	e
	10	8
	-0	#
	21	(_
	61	q
	3	3
	0	→
	55	= :
	22	s
	4	#
	71	i
	57	+ /
	24	u
	2	2
	-0	ß
	73	.)
	64	d
	5	5
	51	r
	1	1
	41	j
	7	7
	45	n
	33	, (
	66	c
	6	6
	63	c _
	54	-
	42	k
	-0	#
	23	t
	-0	#
	31	z
	75	←
	43	l
	0	→
	26	w
	-0	#
	70	h
	0	2
	30	y
	-0	#
	47	p
	-0	#
	50	q
	-0	#
	46	o
	-0	stop
	62	b
	-0	#
	67	g
	-0	#
	11	9
	-0	#
	44	h
	74	↑
	27	x
	-0	#
	25	v
	72	↓
	20	0
	-0	delete

Initializations (temporary)

bc	1
mjp	-1
fc	0
rb	1
fi	fb + 1
ft	fb

Pointers from nt

RND (mach. lang.)
WR (mach. lang.)
PPP

PDP-1

```
rnd,    0
        dap rnx
        lac rn
        shr 1s
        xor (110371
        add (110371
        dac rn
rnx,    jmp
```

TX-0

```
rnd,    sh rnx
        ch (v75102
        lead rn
        qpr 612
        sto rn
rnx,    xx
```

```
wr,     0

        dap wre
wr1,    law i wre
        idx wr
        sad wre
        jmp wr1
wrx,    jap i wre
wre,    0
```

```
wr,     sto wrx
        add (-tze+add)
        sh wr1
        alo a

wr1,    xx      (add)
        spd spd pno
        ado wr1
        lpd
wrx,    xx      1 tze back
        cla
        tra wr1
```

```
ppp,    0
        dap pup
        lio pun
        ppb
        rl 6s
        ppb
        rl 6s
        ppb
pup,    jmp
```

```
ppp,    sbz     pup
        p7h
        p7h
        p7h
pup,    xx
```

Dec. 30

Error handling

```
s1y,
s1z,    error overflow
        goto eh
s1x,    error floating
        goto eh
s1w,    error flexo unc
        goto eh
s1v,    error flexo ert
        goto eh

eh,     step1 nl
        addi be
        trace all, s3x
        load nl
        putback m, (700000
        goto te

s3x,    write eha
        text -/ 2 "Table overflow. -"Stubdivide source program. 2 /
eha,    halt
        goto op

rrz,    write rry
        text -/ 2 "Measure two long. -"Reorange tape 2/
rry,    goto eha

pcz,    error flexo nps
        goto te
```

Pseudos

1 s b rest
2 l c copy
3 e
4 end
5 boss
c bubble
e tower
8 alloc
9 units
a key

ps, goria psw

pr1, set ss, 200000
goto te
pr2, set ss, 400000
goto te
pr3, zero ss
goto te

pra, [halt]
prb, [halt]
prc, [halt]

pr4, ~~halt~~ (end)
load C-1
search pbb, 10, pk2
~~halt~~
goto pf

pr5, set st, 14
goto te
pr6, set st, 30
goto te
pr7, set st, 22
goto te
pr8, set st, 24
goto te

pr9, set a0, 1
sett psw, p9a
goto te
p9a, load x1 ~~store 1x~~
x2 to 1
addi x1
store 3x ~~set il, -1~~

psr, zero a0
goto te

START TAPE

```
ap,     clearo pbb, pbb+7, -1
pf5,    sett riw, pf2
        call ri
ri2,    call rin
        store n1
        call rin
        store n2
        call cr
        zero rij
        goto ri4
ri5,    load rij
        lookup tw
        store chr
        type chr
        stepl rij
ri4,    tles rii, ri5
        type (36
        load n2
        taze rig
        tlesc 11, ri7
rig,    write ri6
        text→ bad→part→count.)/
ri6,    halt
        goto ap
ri7,    move n2, t3
        store np
        goto ri8
ri9,    load t3
        putback pbb, (100000
        stepl t3
ri8,    tlesc 10, ri9
        load n1
        putback pbb, (0
        call wd
        load n2
        call tyd
        write tp
        text→/→parts/
tp,     load n1
        call ppt
        t9

        call blk
        call cr
        zero wf
        zero bc
        set mjp, -1
        goto clu

pf,     zero pfw
        sett riw, pf1
        call ri
        testo pfw, pfa
        write pfh
        text→( bbb→redisappeared.)/
pfh,    halt
pfa,    call rin
        store n1
        call rin
        tael np, pfb
        write pfp
        text→(part→count→did→not→match.)/
pfp,    halt
pfb,    load n1
        call ppt
        load n1
        lookup pbb
        termi pfc
        write pfg
        text→part→duplication.)/
pfg,    halt
pfc,    load n1
        putback pbb, bc

clu,    zero 1u
        zero 3u
        zero ft
        zero ao
        zero tc
        zero ss
        zero bn
        set rb, 1
        store @
        set tim, -1
        set st, 30
        copy nt, kt, 44
        copy kt, nt, 44
        goto s1
```

```
pf1,  lookup tit
      tbl chr, rig
      sett pfx,1
      goto rig

pf2,  putback tit,chr
      goto rig

ppt,  answer ppx
      store t1
      call wo
      write your
      text -> /2 part -> /
prn,  load t1
      call typ
      call blk
      call cr
ppx,  exit

cr,   answer crx
      type (77
crx,  exit

ri,   answer rxr
      zero rii
ri3,  call rt
      store chr
      ftbel (77, rxr
      tbel (36, rit
ri1,  load rii
      goria riw
rig,  step1 ril
      tbesc 100, ri3

rit,  call rt
      tbl (77, rxr
      goto rit
rxr,  exit

rin,  answer rix
      zero num
ria,  call rt
      store dig
      tbze ria
      ftbel (33, riy
      ftbel (77, riy
      tbec (20, rib
      load dig
      tbnl (20, rib
      zero dig
rib   load num
      x10dec
      addi dig
      store num
      goto ria
riy,  load num
rix,  exit

tyc,  answer tyx
      store t1
      tbnz tye
      sett t1 20
tye   type t1
tyx,  exit
```

PSEUDO kcb

```
key,
pva,    call rch
        trze pva
        ftrel (55, pum
        ftrel (57, pus
        trel (54, puf
        call src
        goto s1

pum,    sett pfu, 1
pun,    copy nt, kt, 44
pue,    copy kt, mt, 44
        govia pfu

pus,    sett pfu, put
        goto pub
puf,    sett pfu, pug

pwh,    set psw, pun
        set ac, 1
        goto s1

put,    set acc, 2
        set aci, 4
        set tne, 3
        goto puw

pug,    set acc, -1
        set aci, 3
        set tne, 6

puw,    testb n1, psr
        move tne, ton

pw1,    load ton
        lookup w
        addi acc
        store t4
        load ton
        putback kt, t4
        step ton, 7
        tbse 44, pw1
        istepa n1, 1
        step tne, aci
        tbse 7, puw
        load tne
        subt (7
        store tne
        goto puw
```

Pseudos Rest, Copy Jan 1

```
pvb,    set ao, 1          /rest
        testb mm, pb2
        set psw, psr
        error flexo ilr
        goto te

pb2,    set psw, pb1
        goto te
pb1,    testb irl, pb3
        step1 rl
        addi bc
        tnzc all, s3x
        load 1c
        addi 100
        store t1
        load rl
        store irl
        putback rel, t1

pb3,    step1 bc
        addi rl
        tnzc all, s3x
        load bc
        complement
        putback bar, irl
        istepa rl, 1
        tnz pb3
        goto psr

pvc,    set ao, 2          /copy
        testb mm, co2
i2a,    (set psw, co1
        error flexo ilc)
        goto te
i1a
co1,    set psw, psr
        goto te

co2,    move rl, cbh
        tnze co3
        ftnel bc, co4
        tbs bc, co4
co3,    error flexo blc
        set psw, psr
        goto te
co4,    set psw, co5
        goto te

co5,    load rl
        tnze co6
        ftnel bc, co7
        tbs bc, co7
co6,    error flexo brc
        goto psr

co7,    istepa cbh, 1
co8     load cbh
        step1 bc
        addi rl
        tnzc all, s3x
        load bc
        complement
        putback bar, t4
        step1 cbh
        tbs rl, co8
        goto psr
```

testb pb2

testb mm, pb2

tbs rl, co8

complement
lookup bar
store t4

Jan. 3

PSEUDO PART

```
pre,  test1 pfi, pry
      sett ac, 2
      sett psw, pr1
      goto te
pr1,  move n1, cbh
      sett psw, pr2
      goto te
pr2,  test1 mpt, pra
      move n1, np
prb,  tzee pr2
      tgue 10 pr2
      move n1, t3
      goto ri8
ri9,  load t3
      pwback pbb, 100000
      stop1 t3
ri8,  tlse 10, ri9
      goto prc

pra,  testw n1, np, pr2
prc,  test0 cbh, pr2
      addi -1
      tgue np, pr2
      load cbh
      locking pbb
      termi prd
      write pr3
      text "part duplication2"
pr3,  halt
prd,  load cbh
      pwback pbb, bc
      int bfi, 1

      set mpt, 1
      goto psr

pry,  write pr9
      text "extra part"
pr9,  sett psw, i2a
      goto te

pr2,  write pr8
      text "bad prefix"
pre,  halt
      goto pf
```

START of tape

tp uses t1

```
ap,     clear  pbb, obb+7, -1
        zero   mpt
        set    ft, fb
pf,     call   tp
pg,     store  chn
        load   ft
        putback f, chn
        testul chn (p1, pg1
pg1,    skpl   ft
        testl  (f, pg
        halt
hap,    goto   ap

pfr,    zero   1u
        zero   3u
        sett   iw, -1
        zero   ft
        set    fi, 1
        zero   qo
        zero   te
        zero   ss
        zero   bn
        set    rb, 1
        store  tbc
        set    tun, -1
        set    st, 30
        goto   pum

xp,     set    pfu, pfr
        set    emp, fb
        set    te, -1
        goto   enf

        zero   bc
        zero   ben
        set    wsp, -1
        store  wl

INSERT

/copies mt → kt → mt, goes to s1
```

Φ2

TABLES

nos: multiple - string 2D table

npo: depth in nos

nmu: points in not

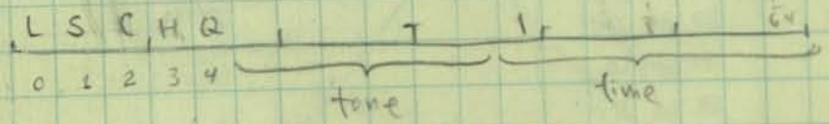
npl points in bar

ton: tone format

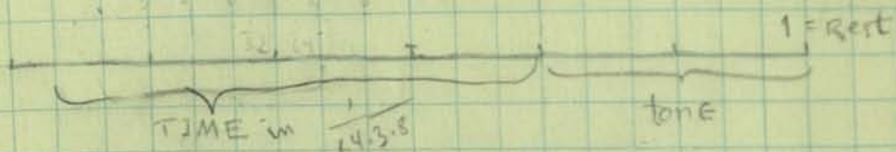
Intermediate Tape Format

400 blank
1 wd no. of notes = m
m wds notes
1 wd chksum of NOTES only
6 blank
1 wd no. of bars = n
n wds bar entries
1 wd chksum of BAR ENTRIES
300 blank

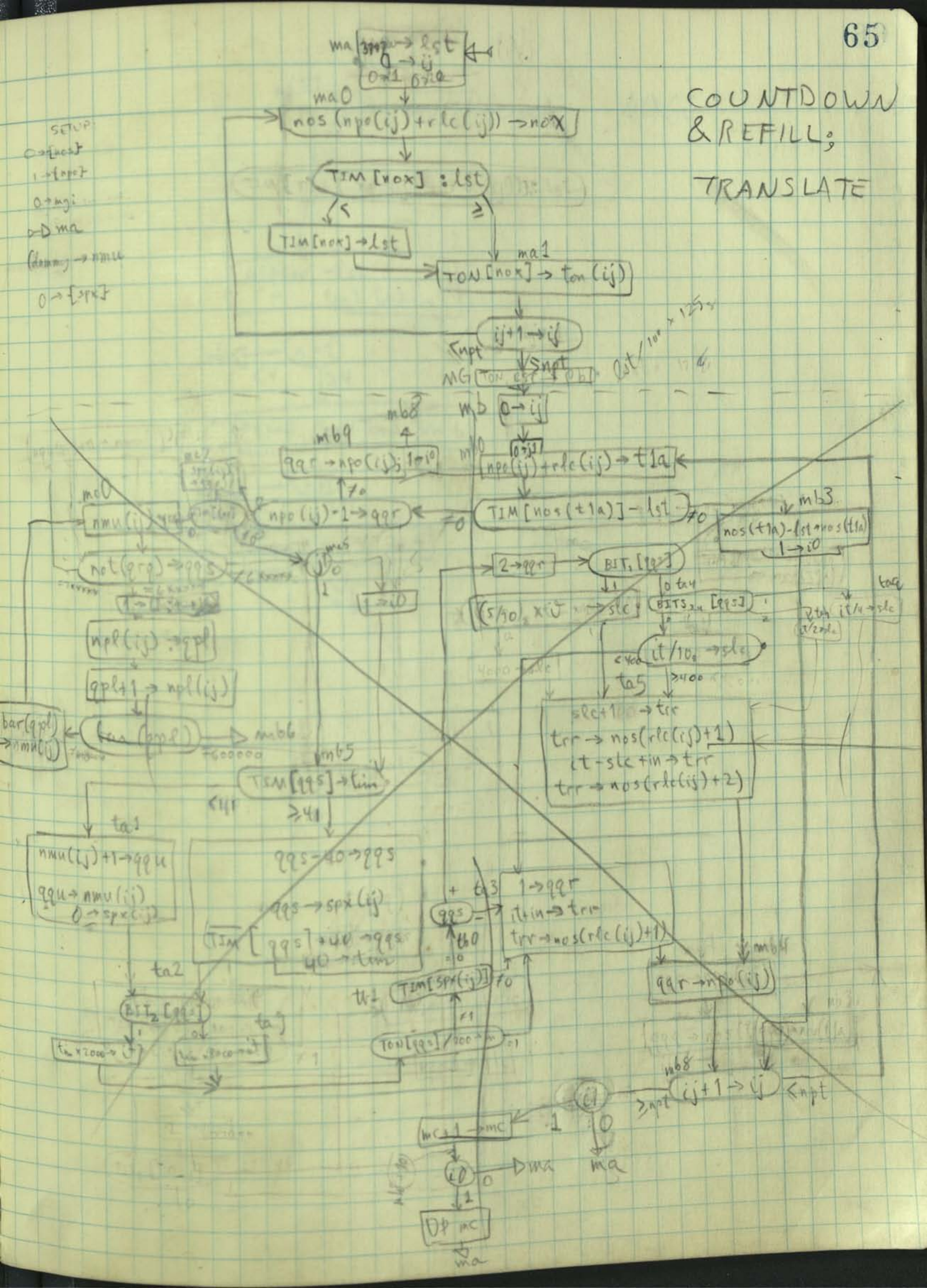
FORMAT before TRANSLATION

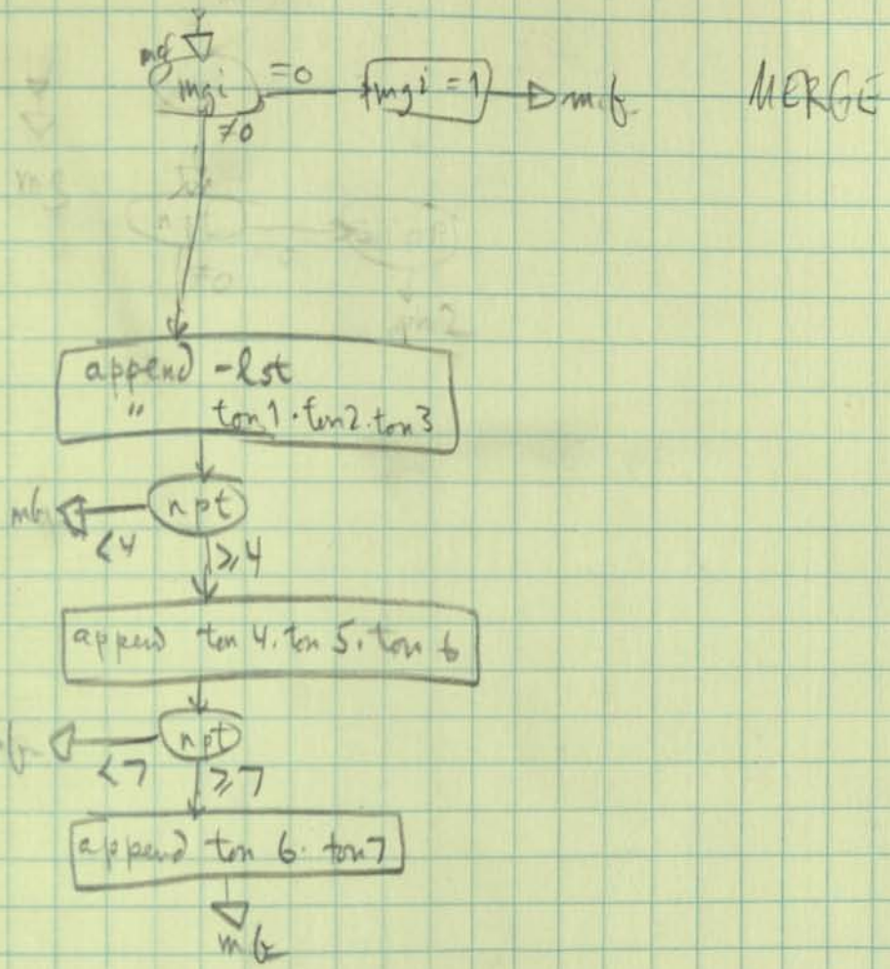


FORMAT after TRANSLATION



COUNTDOWN & REFILL; TRANSLATE




mg
mgi
=0
mgi = 1
mg
MERGE
≠0
append -lst
" ton1 · ton2 · ton3
npt
mb
<4
≥4
append ton4 · ton5 · ton6
npt
<7
≥7
append ton6 · ton7
mg

List tests & mechanize or flush
Generalize

Watch testing

Statistics now

1 DDT
EXPENSIVE

2 MULT PREC

dzm y
dac y
jms y
lac y
xor y
adc y
add y
xct y
isp y / iso y
and y
jmp y
iot
opr

1. IOT
2. OPR; skip before sft?