

SUBROUTINES

,Divide + Multiply test using random numbers

,9/25/61

,B.G. S.L.

```
be1      dzm ctn1 ,no. of goods
         dzm ctn2 , " " "
         dzm ctn3 , " " over-flows
         dzm ctn4 , " " " "
         lac rand1 ,initilize random no. gen.
         dac rd1
         lac rand2 , " " " "
         dac rd2
p1      clf 6
         szo ,clear overflow
         nop
         jsp abc ,put rand no. in a,b,c
         rar s1 ,AC holds b
         xor a
         sma ,is bit 17 of b equal to 0 of a
         jmp p2
         lac b
         xor rc1
         dac b ,bit 0 of a equals bit 17 of b
p2      szs 60
         jmp muldiv ,multiply + divide subroutine
         lac a
         lio b
         dis c ,high speed divide
         jmp divovflo
         dac quot
         dio rem
p3      mus c ,high speed multiply
         szs 20
         jmp p2
         spa
         stf 16 ,remember prod negative
         dac hprod
         dio lprod
         lac lprod
         szo
         nop ,clear overflow from divd subroutine
         rar s1
         add rem ,add remainder to low product
         szo ,is there a carry into high product
         jmp p4
         ral s1
         dac t1
         jmp p5
p4      xor t2 ,fix up sign bit of low product
         ral s1
         dac t1
         law 1
         szf 6 ,sub one if hprod is negative
         cma
         add hprod
         dac hprod ,lprod + rem carry added to hprod
         lac t1
p5      sas b
         jmp errorone ,low product + remainder not equal to b
         lac hprod
         sas a
         jmp errortwo ,high product not equal to a
         clf 6
         idx ctn1
         sma
```

```

      jmp p1
      dzm ctn1
      idx ctn2 ,count sucessful matches
      jmp p1
muldiv  lac a
      lio b
      jda dvd ,use divide subroutine
      lac c
      jmp divovflo
      dac quot
      dio rem
      lac quot
      jda mpy ,use multiply subroutine
      lac c
      jmp p3
divovflo  idx ctn3
      sma ,count no of overflows

      idx ctn4
      dzm ctn3
      jmp p1
a 0
b 0
c 0
quot 0
rem 0
hprod 0
t1 0
lprod 0
ctn2 0
ctn1 0
ctn4 0
ctn3 0
t2 400000
t3 2
t4 lio t2 ,stop code for type out
t5 0
rc1 1
cr 77
rand1 742335
rand2 125131
rd1 0
rd2 0
errorone lio rc1 ,print out errors
      jmp errortwo + 1
errortwo lio t3
      tyo '
      law a
      dap p12
p11 lio t2
      tyo '
p12 lio
      law ' 6
      jda print
      idx p12
      sas t4
      jmp p11
      lio cr
      tyo '
      szs 20
      jmp p2
      szs 10
      jmp be1
      jmp p1
print 0
      dio t5

```

```

dap p10
p7   cla
      lio t5
      rcl s3
      sza '
      lio t3
      lac t5
      rcl s3
      dac t5
      tyo '
      isp print
      jmp p7
p10   jmp
abc   dap out ,get random number
      jsp random
      dac a
      jsp random
      dac c
      jsp random
      dac b
out   jmp
random dap exit ,random num generator
      lac rd2
      lio rd1
      rcr s7
      xor rd1
      lio rd2
      dac rd2
      dio rd1
exit  jmp
      jmp be1 end .
~

```

org 100

PUNCH TEST

```

m      lac k      ,ppa ', setup punch 0, general test
      dac a + 1
      dac a + 5
      dac c + 1
      dac c + 3
      dac c + 7
      dac c + 11
      lac k + 1      ,ppb '
      dac e + 1
      jmp b
n      lac k + 2      ,ppa ' c1, setup punch 1
      dac a + 1
      dac a + 5
      dac c + 1
      dac c + 3
      dac c + 7
      dac c + 11
      lac k + 3      ,ppb ' c1
      dac e + 1
b      lac j + 1      , -2 reset a ctrs
      dac h + 4
      lac j      , -12
      dac h + 1
      dac h + 3
a      lio h      ,377, 10 lines all holes
      0
      isp h + 1      , -12
      jmp a
      lio h + 2      ,777400, 10 lines no holes
      0
      isp h + 3      , -12
      jmp a + 4
      isp h + 4      , -2
      jmp b + 2
d      lac h + 2      ,reset c ctrs
      dac h + 6
      dac h + 7
c      lio h + 5      ,0, o,n,, on+1, etc.
      0
      lio h + 6      ,777400
      0
      isp h + 6
      jmp c
      lio h      ,377, n.,377, n + 1, etc.
      0
      lio h + 7      ,777400
      0
      isp h + 7
      jmpc + 6
f      lac h + 11      , -100, reset e ctrs
      dac h + 10
      lac h + 5      ,0
      dac h + 12
e      lio h + 12      ,0 init, bin 0-77
      0
      lac h + 12
      add h + 13      ,10000
      dac h + 12
      isp h + 10
      jmp e
      szs 10
      jmp aa
      jmp e + 7

```

```

h      377      ,constants
      0
      777400
      0
      0
      0
      0
      0
      0
      - 100
      0
      10000
j      - 12
      - 2
org 220
g      lac k      ,0a, tw mode
      jmp g + 7
      lac k + 1      ,0b
      jmp g + 7
      lac k + 2      ,1a
      jmp g + 7
      lac k + 3      ,1b
      dac r
p      dac r + 2
      lac k + 4      ,-23 setup line ctr
      dac k + 5
      lac k + 6      ,-60k stop delay 250ms
      dac k + 7
      lat
      dac k + 10
      lio k + 10
r      0      ,punch rt half if a, lt half if b
      rir s9
      0      ,punch lt half if a, rt half if b
      isp k + 5      ,-23 line count
      jmp p + 7
      isp k + 7      ,-60k stop delay
      jmp r + 5
      jmp p + 1
k      ppa '      ,constants
      ppb '
      ppa ' c1
      ppb ' c1
      - 23
      0
      - 60000
      0
      0
aa     lac ah      ,100 locate data, general test read check
      dio ah + 1
      sad ah + 1
      jmp aa + 1
ac     lac j + 1      ,-2 reset ab and ad
      dac ah + 7
      lac ah + 4      ,-12
      dac ah + 3
      dac ah + 6
ab     lac ah + 2      ,377, check 10 lines all holes
      xor ah + 1
      sza
      jda bd      ,error
      isp ah + 3      ,-12
      jmp ab + 7
      jmp ad      ,check next 10
      rpa '

```

```

        dio ah + 1
        jmp ab
ad      rpa '      ,check 10 lines no holes
        dio ah + 1
        lac ah + 5      ,0
        lor ah + 1
        sza
        jda bd      ,error
        isp ah + 6      ,-12
        jmp ad
        isp ah + 7      ,-2
        jmp ad + 13      ,check next 10, 377
        jmp an
        rpa '
        dio ah + 1
an      jmp ac + 2
        lac ah + 5      ,0 reset for 0,n
        dac aj
        dac ah + 6
        lac aj + 1      ,-377
        dac ah + 3      ,pr ctr
        lac j + 1      ,-2
        dac ah + 7
        jmp ae      ,check 0,n
ae      rpa '
        dio ah + 1
        lac aj      ,0
        xor ah + 1
        sza
        jda bd
        rpa '
        dio ah + 1
        lac ah + 6      ,0 init
        xor ah + 1
        sza
        jdl bd      ,error
        isp ah + 3      ,-377
        jmp ae + 17
        jmp af
        idx ah + 6      ,compare word
        jmp ae
af      isp ah + 7      ,-2
        jmp af + 3
        jmp af + 12
        lac ah + 2      ,377 reset for 377, n
        dac aj
        lac ah + 5      ,0
        dac ah + 6
        lac aj + 1      ,-377
        dac ah + 3      ,pr ctr
        jmp ae      ,check 377,n
        lac aj + 2      ,-100, reset for binary
        dac aj + 3
        lac aj + 4      ,200 compare word
        dac aj + 5
ag      rpa '      ,check binary
        dio ah + 1
        lac aj + 5      ,200 init
        xor ah + 1
        sza
        jda bd      ,error
        isp aj + 3      ,-100
        jmp ag + 11
        jmp ag + 13
        idx aj + 5
        jmp ag

```

```

lac ah      ,100 check feed
rpa '
dio ah + 1
sad ah + 1
hlt         ,test complete
jda bd      ,error
ah 000      ,constants
0
377
0
- 12
0
0
0
0
aj 0
- 377
- 100
0
200
0
org 450
ba lac ah      ,100, locate data
rpa '
dio ah + 1
sad ah + 1
jmp ba + 1
bb lat         ,check data, rt alpha
and ah + 2      ,377
xor ah + 1
sza
jda bd      ,error
rpa '
dio ah + 1
bc lat         ,check data lt
sar s9
and ah + 2      ,377
xor ah + 1
sza
jda bd      ,error
rpa '
dio ah + 1
jmp bb
org 500
be lac ah      ,100 locate data
rpa '
dio ah + 1
sad ah + 1
jmp be + 1
bf lat         ,check data , lt bin
and bk      ,770000
rar s6
rar s6
add aj + 4      ,200
xor ah + 1
sza
jda bd      ,error
rpa '
dio ah + 1
bg lat         ,check data, rt bin
and bk + 1      ,770
rar s3
add aj + 4      ,200
xor ah + 1
sza
jda bd      ,error
rpa '

```

```

    dio ah + 1
    jmp bf
bk    770000
    770
bd    0
    dap bd + 4      ,error routine
    lacbd
    hlt
    jmp
org 0
bj    0
    jmp end
```

```

,TAPE CONTROL PROGRAM S. L.
,calling sequence
,law or lac command
,jda tape
,initial address back or foward
,final address " " "
,HLT non normal return
,HLT normal return
opd msm 720073
opd mwc 720071
opd mrc 720072
opd mec 720034
opd mcb 720070
org 7000
tape 0 ,command
dap → + 1
q100 lac
dap k1
idx q100
lac ' q100
dap k2
idx q100
dap k3
lio tape
rir s8
spi ,operate bit off
jmp p104
lac t105
sza ,tape is sopped
lio tape
msm
cla
p102 mec ,strobe status
jmp ' k3
p104 lac tape
and t111
dac t3
lac t105 ,what unit number
dac t2
and t111
sas t3 ,was last operation continue
jmp p60
mec
spi ,is present unit busy
jmp p103
p60 lio tape
msm ,start up tape transport
p103 dzm t105
dzm t122 ,flag 4
law p132
dap p131
law p131
dap p134
law q6
dap p122
dap q7 + 1
dap q7 + 3
dap q10 + 2
law 6
dap p32
law q15
dap q13 - 4
law p44
dap p133

```

```

law p50
dap p125
law p52
dap p130
law q1
dap q13
mcb
lio tape
ril s6
spi ,early complete different unit
p113 jmp p105
ril s1
spi ,its your gap
p114 jmp p106
ril s1
spi ,read check
p115 jmp p107
ril s1
spi ,space back or fow ard
p116 jmp p110
ril s2
spi ,rewind
jmp p137
ril s1
spi ,foward or reverse
p117 jmp p111
ril s1
spi ,read or write
jmp p112
p105 jmp p120
law p122
dap p123
p106 jmp p113
lac tape
dac t105
law p122
dap p133
dap p125
dap p130
p107 jmp p114
law rdck
dap p134
p110 jmp p115
law space
dap p134
law p134
dap p133
dap p125
lac k1
cma
dac t110
p137 jmp p116
idx k3
dzm bk
cli cla
msm
mec
p111 jmp ' k3
law bread
dap p131
p112 jmp p117
lac t2
sza '
jmp → + 4
law write + 2
da p134

```

```

p120      jmp p120
          law write
          dap p134
          lac k1
          dap q31
          law ' 1
          add k1
          dap q6
          dap q15
          lac k2
          dap q33
          dap q11
          dap t103
          cli 7
p134      jmp p131
p131      jmp p132
p132      lac t107
          dac q7
          lac q32 + 1
          dac q2
          dac q4
          lac q30
          dac q5
          idx bk
          jsp read
p133      jmp p44      ,your gap
p44      lac t113      ,fixed delay for eob
          jda delay
p45      lio t3
          msm
p123      jmp p46      ,early complete different
p46      lac t114      ,fixed stop delay
          jda delay
p122      lac q6
p32      szf 6
          jmp p56
          lac t122
          sza '
          jmp → + 3
          stf 4
          jmp p56
          idx k3
          lac bk
          mec
p56      jmp ' k3
          lac ' p122
          mec
bread     jmp ' k3      ,non-normal return
          lac q6
          dap q11
          dap t103
          lac k2
          dap q6
          dap q15
          lac t104
          dac q7
          lac t106
          dac q2
          dac q4
          dac q5
          law ' 1
          add bk
          dac bk
          stf 6
          jsp read
p125      jmp p50      ,your gap

```

```

p50      lac t115 ,delay back
         jda delay
         jmp p45
write    lac t116
         jda delay
         idx bk
         law q31
         dap p122
         jmp q30
q35      lac t117
         jda delay
         mcb
p130     jmp p52 ,your gap
p52      lac t120 ,stopping delay
         jda delay
         jmp p45
space    law p22
         dap q13
         isp t110
         jmp p131
         law p44
         dap p133
         law p50
         dap p125
         law 2000
         dap p32
         jmp p131
rdck     law q6
         dap q13 - 4
         law q15
         dap p122
         dap q7 + 1
         dap q7 + 3
         dap q10 + 2
         jmp p131
q30      nop
         law ' 2
         dac t2
q31      lio
q32      mwc
         ril s6
         cla
         jda delay
         isp t2
         jmp q32
         nop
         mwc
         nop
         idx q31
         sad q33
         jmp q35
         jmp q30
read     dap p31
         mcb
         lac t112 ,delay before error miss
         dac t2
p2       szf 2
         jmp → + 4
         isp t2
         jmp p2
         jmp q14 ,tape has no characters
         law t123
         cli
         dap
         mrc
         cla

```

	nop
q13	jmp q1
q1	szf 2
	jmp q2
	szf 2
	jmp q2
	szf 2
	jmp q2
	szf 2
	jmp q2
	szf 2
	jmp q2
	szf 2
	jmp q2
	szf 2
	jmp q2
	szf 2
	jmp q2
	szf 2
	jmp p22
q2	rll s6
q7	jmp q10
	add q6
	mrc
	dac q6
q10	jmp q3
	lac q6
	mrc
	idx q6
q3	jmp q3
	sad q11
	jmp p26
	szf 2
	jmp q4
	szf 2
	jmp q4
	szf 2
	jmp q4
	szf 2
	jmp q4
	szf 2
	jmp q4
	szf 2
	jmp q4
	szf 2
	jmp q4
	szf 2
	jmp q4
	szf 2
	jmp q4
	szf 2
	jmp q4
	szf 2
	jmp q14
q4	rll s6
	mrc
q5	nop
q6	dio
	cli clf 6
	szf 2
	jmp q12
	szf 2
	jmp q12
	szf 2
	jmp q12
	szf 2
	jmp q12
	szf 2

```

        jmp q12
        szf 2
        jmp q12
        szf 2
        jmp q12
        szf 2
        jmp q12
        jmp q14
q12     lac t123
        mrc
q15     sas
        jmp p26 - 1
        jmp q1
q14     law ' 0
        dac t2
        szf 2
        jsp q16
        isp t2
        jmp q14 + 2
        jmp p26
q16     dap → + 3
        stf 6
        mrc
        jmp
p22     szf ' 6      ,is it eob
        jmp q14
        mrc
        clf 7
        jmp p2
        stf 6
p26     szf 4
        idx t122
        law ' 14      ,delay after character
        dac t2
        mrc
p27     szf 2
        jmp p26 + 2
        isp t2
        jmp p27
        cli clf 4
p31     jmp
delay   0
        dap exit
        isp delay
        jmp → - 1
exit    jmp
q11     dio
q33     lio
t2      0      ,index counter
t3      0      ,unit no + temp stor
k1      0      ,initial address
k2      0      ,final address
k3      0      ,non norm return
t7      0      ,address equal
t103    sas
t106    rir s6
t104    law ' 1
t105    0      ,continue memory
t107    jmp q10
t110    0      ,space count
bk      0      ,block count
t111    000003 ,unit mask
t112    - 764  ,begining delay onread mis char
t113    - 102  ,foward delay read before stop

```

t114	- 306	,final stop delay read + write
t115	- 412	,back read delay before stop
t116	- 424	,write start delay
t117	- 13	,eob delay write
t120	- 524	,write delay before stop
t122	0	,flag 4
t123	0	,storage for read check
end		

```

,anytape duplicator verifier
org 3000
    cla
    dap zero
zero    dzm
        idx zero
        sas limit          ,limit is 340400
        jmp zero
        hlt          ,set master then contin
leader  rpa '
        rcr s9
        rcr s9
        sas stopcode
        jmp leader
        rcr s9
        rcr s9
        szs 20          ,on is no punching
        jmp setreadbloc
        law punchbloc
        dap punchorread
        law 1000
        dap block
        jmp patch1
readbloc law 1000
        dap block
        rpa '
        rcr s9
        rcr s9
        sad stopcode
        jmp patch2
        dap histo
block   dac
histo   idx
        idx block
        sas limit1          ,limit1 is 241100
        jmp readbloc + 2
punchorread jmp
punchbloc  law 1000
        dap block1
        lac block
        dap limit2          ,limit2 is 22xxxx
block1   lio
        ppa '
        idx block1
        sas limit2
        jmp block1
        sad limit3          ,limit3 is 221100
        jmp readbloc
loadcopy hlt          ,set copy then contin
        law 0400
        dap zero1
zero1    dzm
        idx zero1
        sas limit4          ,limit4 is 341000
        jmp zero1
leader1  rpa '
        rcr s9
        rcr s9
        sas stopcode
        jmp leader1
        jmp read + 5
read     rpa '
        rcr s9

```

```

rcr s9
sad stopcode
jmp patch3
add constant      ,constant is 000400
dap histo1
histo1   idx
        jmp read
compare  law 0000
        dap masterhisto
        law 0400
        dap copyhisto
masterhisto   lac
copyhisto     sas
        jmp error
indexer   idx masterhisto
          idx copyhisto
          sas limit5      ,limit5 is 521000
          jmp masterhisto
          jmp loadcopy
error     lio save3
          tyo '      ,cr
          lio lc
          tyo '      ,lc
          lio sam
          tyo '      ,m
          rir s6
          tyo '      ,a
          rir s6
          tyo '      ,s
          cli
          tyo '      ,space
          lac masterhisto
          and save3
          jda type
          cli
          tyo '      ,space
          lac ' masterhisto
          jda type
          lio save3
          tyo '      ,cr
          lio poc
          tyo '      ,c
          rir s6
          tyo '      ,o
          rir s6
          tyo '      ,p
          cli
          tyo '      ,space
          lac copyhisto
          and save3
          jda type
          cli
          tyo '      ,space
          lac ' copyhisto
          jda type
          lio save3
          tyo '      ,cr
          szs 10      ,on to contin despite err
          jmp indexer
          hlt          ,if ss1 dn end prog
trailer  szs 20
          jmp loadcopy
          jmp punchbloc
setreadbloc   law readbloc
          jmp readbloc - 4
limit        340400

```

stopcode	000113
limit1	241100
limit2	220000
limit3	221100
limit4	341000
constant	000400
limit5	521000
poc	474663
lc	000072
sam	226144
save3	000777
type	0

	dap exit
	dzm temp1
	law ' 6
	dac temp
	lac type
	cli
	rcl s3
	dac type
	cla
	rcr s3
	sza '
	jmp leading
	rcl s3
	idx temp1
go	tyo '
	isp temp
	jmp type + 5
exit	jmp
leading	sas temp1
	jmp twenty
	law ' 1
	sas temp
	jmp go
twenty	llo two
	jmp go
temp	0
temp1	0
two	000020
patch1	rcr s9
	rcr s9
	jmp readbloc + 7
patch2	dap n
	dac block
n	idx ..
	idx block
	jmp trailer
patch3	add constant
	dap m
m	idx ..
	jmp compare
	jmp end

...multiply subroutine

...

xsy mpy

imp':b: loc
dap a

a: xct
jda g
lac b
idx a
rir 1
rcr 9
rcr 9
jmp'a

e: loc ... temporary storage

mpy':g: loc
dap f
lac g
spa
cma
rcr 9
rcr 9

f: xct
spa
cma
dac e
cla

z eoc mus

z e,z e,z e,z e,z e,z e,z e,z e,z e,z e,z e,z e,z e,
z e,z e,z e ...17mus e

dac e
xct f
xor g
sma
jmp h
lac e
cma
rcr 9
rcr 9
cma
rcr 9
rcr 9
dac e

h: idx f
lac e
jmp'f

fin

...decprint subroutine, routine to print decimal
...July 11, 1961

```
...  
decprint',      dap return  
    law teny  
    dap tenz    ...set the nonprint trap  
    dzm value   ...reset value  
    law'5  
    dac position    ...reset position  
    law a  
    dap tenj    ...reset end trap  
    rcl 9  
    rcl 9  
    sma        ...check for minus  
    jmp b  
    cma  
    dac store  
    law c  
    dap printit    ...set trap to print minus sign  
    jmp set  
b.    dac store  
    cli  
    xct command    ...print space since no minus sign  
set:   law store  
    add position  
    dap d        ...next number to subtract  
form:  lac store  
d:     sub  
    spa  
    jmp print  
    dac store  
    idx value  
    jmp form  
print: add'd  
    dac store    ...restore after going negative  
    lac value  
e:     sza  
    jmp printit  
printzero: law 20  
tenz:   jmp teny  
tenx:   jmp printit  
teny:   cli  
    xct command    ...trap to print no leading zeros  
    jmp tenb  
printit: jmp tenk  
c:     lio minus  
    xct command  
tenk:   rcl 9  
    rcl 9  
    xct command  
    dzm value  
    law tenk  
    dap printit  
    law tenx  
    dap tenz    ...unset nonprint trap  
tenb.   isp position  
    jmp set  
    law tenx  
    dap tenz  
tenj:   jmp → + 1  
a:     law return  
    dap → - 2  
    lac store  
    jmp e
```

```
loc 303240
loc 23420
loc 1750
loc 144
loc 12
store: loc
value: loc
position: loc
return: jmp
minus: bci -.
command: tyo'
fin.
```

,binary punch and load package modified 19JAN61 for read in check and
 ,lo and mid version (loader first)
 ,binary loader with check feature added
 ,19 JAN 61 BF

```

    org 1
a    dzm sum      ,clear suegister
    jsp rbs      ,read binary word and sum
    dac b
    dap c
    spa
    jmp e        ,negative ac      -      assumed jump
    lac a + 2
    szs 20
    lac c-p
    dip c
    jsp rbs      ,read length
    add c        ,compute end
    dac tsl
    jsp rcs      ,read check sum and check
d    jsp rbs      ,read binary word and sum
c    ..          ,dac in memory or sad with memory
    jmp → + 3
    lac ' c
    hlt
    idx c        ,index store or check address
    sas tsl      ,checks for end of block
    jmp d        ,store or check next word
    jsp rcs      ,end of block read check sum and check
rcjmp
e    jmp a
    jsp rcs
    szs 10       ,ss 1 off jump to address; set - halt
    hlt
b    jmp ..
rbw  dap f        ,read binary word subroutine
    rpb '
    dio bwd
    lac bwd
f    jmp ..
rbs  dap g        ,read binary word and sum subroutine
    jsp rbw
    add sum
    dac sum
    lac bwd
g    jmp ..
rcs  dap h        ,read check sum and check subroutine
    jsp rbw
    lac sum
    sas bwd
    hlt
    dzm sum
h    jmp ..
c-p  sad ..
sum  ,            contains sum
bwd  ,            contains binary word
tsl  ,            temporary storage
rim  jsp longlead
    lac rcdio
    dac bwd
15q  lio bwd
    jsp pbw
    lio ' bwd
    jsp pbw
    idx bwd
    sas rcendload

```

```

        jmp 15q
        lio rcjmp
        jsp pbw
        jmp pbt
,control and punch section
'scontrol   dzm return?
control     lac jump?
           sza
           jmp pjp
           lac return?
           sza
return      jmp ..
clear       dzm sum
next        cli clf 1 cla
           szf 1
           jmp → + 2
           jmp → - 2
           tyi
           rcr s9
           rcr s9
           sad rc>
           jmp clear
           lio sum
           sad rc-b
           jmp begin
           sad rc-l
           jmp length
           sad rc-f
           jmp final
           sad rc-r
           jmp rim
           sad rc-p
           jmp pbt
           sad rc-j
           jmp jump
           sad rc-s
           jmp a
           ril s3
           rcr s3
           ril s3
           dio sum
           jmp next
begin       dio inl
           dzm jump?
           jmp clear
length      dio len
           jmp clear
final       idx sum
           sub inl
           dac len
           jmp clear
jump        dio jpl
           idx jump?
           jmp control + 3
rc-l        43
rc-f        66
rc-r        51
rc-p        47
rc-b        62
rc-j        41
rc-s        22
rc>         56
return?
jump?
,binary format punch routinepbt

```

```

        jsp ptf      ,initial tape feed

```

```

        ,llo starting address
        llo inl
        dio bwd
        jsp pws      ,punch location block containing initial location of
        llo len      ,program block, length of block and sum
        jsp pws
        dac fnl
        llo sum
        jsp pbw

,
jsp ptf      ,punch program block with check sum at end
b1          llo ' bwd
           jsp pws
           idx bwd
           sas fnl
           jmp b1
           llo sum
           jsp pbw,
exit        jmp control

,
pbw         dap a1      ,punch binary word subroutine
           xct punch-b
           ril s6
           xct punch-b
           ril s6
           xct punch-b
a1          jmp ..

,
pws         dap d1
           dio tsl,    punch word and sum subroutine
           jsp pbw
           lac tsl
           add sum
           dac sum
d1          jmp ..

,
pjp         jsp ptf      ,punch jump pair subroutine
           lac jpl
           dap jpx
           llo jpx
           jsp pbw
           llo jpx
           jsp pbw
           jsp longlead
           dzm jump?
           jmp exit

,
ptf         dap jpx
           llo lead
           dzm sum
h1          xct punch-a
           idx sum
check       sas tfl
           jmp h1
           dzm sum
jpx         jmp ..

,
lead        100
tfl         12
tfl12       340
ts2         ,temporary storage 2
inl         ,initial loaction stored here
fnl         ,final location stored here
len         ,block length stored here
jpl         ,jump location stored here
rcdio       dio a
rcendload   dio 15a + 12

```

```

loglead  dap lr
         idx check
         jsp ptf
         law tfl
         dap check
lr        jmp ..
punch-a   ppa '
punch-b   ppb '

```

,routine for program control of bin p+l package

```

pb        ..          ,beginning of block stored here
         dzm rim?
         dap return
         idx return?      ,set control to return
         lac pb
         dac inl
         dio len
         lac rim?
         sza
         jmp rim
         jmp pbt
pbr        ..
         dap return
         idx rim?
         lac pbr
         dac pb
         jmp pb + 3
rim?      dap return
pj        idx return?
         jmp jump
jmp end

```

,binary punch and load package modified 19JAN61 for read in check and

```
,
scontrol  org 7500
control   dzm return?
          lac jump?
          sza
          jmp pjp
          lac return?
          sza
return    jmp ..
clear     dzm sum
next      cli clf 1 cla
          szf 1
          jmp → + 2
          jmp → - 2
          tyi
          rcr s9
          rcr s9
          sad rc>
          jmp clear
          lio sum
          sad rc-b
          jmp begin
          sad rc-l
          jmp length
          sad rc-f
          jmp final
          sad rc-r
          jmp rim
          sad rc-p
          jmp pbt
          sad rc-j
          jmp jump
          sad rc-s
          jmp a
          ril s3
          rcr s3
          ril s3
          dio sum
          jmp next
begin     dio inl
          dzm jump?
          jmp clear
length    dio len
          jmp clear
final     idx sum
          sub inl
          dac len
          jmp clear
jump      dio jpl
          idx jump?
          jmp control + 3
rc-l      43
rc-f      66
rc-r      51
rc-p      47
rc-b      62
rc-j      41
rc-s      22
rc>       56
return?
jump?
```

,binary format punch routine

```

pbt      jsp ptf      ,initial tape feed
         lio inl      ,lio starting address
         dio bwd
         jsp pws      ,punch location block containing initial location of
         lio len      ,program block, length of block and sum
         jsp pws
         dac fnl
         lio sum
         jsp pbw

,
jsp ptf   ,punch program block with check sum at end
b1        lio ' bwd
         jsp pws
         idx bwd
         sas fnl
         jmp b1
         lio sum
         jsp pbw

,
exit      jmp control

,
pbw        dap a1      ,punch binary word subroutine
         xct punch-b
         ril s6
         xct punch-b
         ril s6
         xct punch-b
a1         jmp ..

,
pws        dap d1
         dio tsl,     punch word and sum subroutine
         jsp pbw
         lac tsl
         add sum
         dac sum
d1         jmp ..

,
pjp        jsp ptf      ,punch jump pair subroutine
         lac jpl
         dap jpx
         lio jpx
         jsp pbw
         lio jpx
         jsp pbw
         jsp longlead
         dzm jump?
         jmp exit

,
ptf        dap jpx
         lio lead
         dzm sum
h1         xct punch-a
         idx sum
check      sas tfl
         jmp h1
         dzm sum
jpx        jmp ..

,
lead       100
tfl        12
tfl12      340
ts2         ,temporary stoage 2
inl         ,initial loaction stored here
fnl         ,final location stored here
len         ,block length stored here

```

```

jpl      ,jump location stored here
rcdio    dio a
rcendload      dio 15q + 13
longlead  dap lr
          idx check
          jsp ptf
          law tfl
          dap check
lr        jmp ..
punch-a   ppa '
punch-b   ppb '
',binary loader with check feature added
,19 JAN 61      BF
          org 7700
a          dzm sum      ,clear sum register
          jsp rbs      ,read binary word and sum
          dac b
          dap c
          spa
          jmp e        ,negative ac      -      assumed jump
          lac a + 2
          szs 20
          lac c-p
          dip c
          jsp rbs      ,read length
          add c        ,compute end
          dac tsl
          jsp rcs      ,read check sum and check
d          jsp rbs      ,read binary word and sum
c          ..          ,dac in memory or sad with memory
          jmp → + 3
          lac ' c
          hlt
          idx c        ,index store or check address
          sas tsl      ,checks for end of block
          jmp d        ,store or check next word
          jsp rcs      ,end of block read check sum and check
rcjmp     jmp a
e          jsp rcs
          szs 10      ,ss 1 off jump to address; set - halt
          hlt
b          jmp ..
rbw        dap f      ,read binary word subroutine
          rpb '
          dio bwd
          lac bwd
f          jmp ..
rbs        dap g      ,read binary word and sum subroutine
          jsp rbw
          add sum
          dac sum
          lac bwd
g          jmp ..
rcs        dap h      ,read check sum and check subroutine
          jsp rbw
          lac sum
          sas bwd
          hlt
          dzm sum
h          jmp ..
c-p        sad ..
sum        ,          contains sum
bwd        ,          contains binary word
tsl        ,          temporary storage

```

```

rim      jsp longlead
         lac rcdio
         dac bwd
15q      lio bwd
         jsp pbw
         lio ' bwd
         jsp pbw
         idx bwd
         sas rcendload
         jmp 15q
         lio rcjmp
         jsp pbw
         jmp pbt
         jmp scontrol
,
         org 7500 - 25
,routine for program control of bin p+l package
pb      ..      ,beginning of block stored here
         dzm rim?
         dap return
         idx return?      ,set control to return
         lac pb
         dac inl
         dio len
         lac rim?
         sza
         jmp rim
         jmp pbt
pbr      ..
         dap return
         idx rim?
         lac pbr
         dac pb
         jmp pb + 3
rim?
pj      dap return
         idx return?
         jmp jump
jmp end

```

```

ext loader      org loader
                dzm sum      ,clear sum register
                jsp rbs      ,read binary word and sum
1a             dac 1c        ,1 nititalize load loop poointer
                spa
                jmp 1e        ,neagative ac - assumeded jump
                lac 1a        ,get dac instruction
                szs 20        ,load or compare?
                lac 1j        ,compare - fetch sad
                dip 1c        ,initialize instruction in load loop
                jsp rbs      ,read leangth
                add 1c
                dac fnl      ,compute end ck
                jsp rcs      ,read check sum and check
1d             jsp rbs      ,read binary word and sum
1c             ..          ,dac or sad
                jmp → + 3      lac ' 1c ,tape differend so shoe memory ■
                hlt          ,comparison error
                idx 1c        ,index store or check address
                sas fnl      ,check for end of block
                jmp 1d
                jsp rcs      ,end of block - read check sum
                jmp loader
1e             jsp rcs      ,jump block so read check sum
                szs 10        ,ss1 off - jump to address; set - halt
                hlt
                xct 1c        ,do the jump
rbw            dap r1        ,read binary word subroutine
                rpb '
                dio bwd
                lac bwd
ext r1         jmp ..
rbs            dap r2        ,rbw and sum subroutine
                jsp rbw
                add sum
                dac sum
                lac bwd
ext r2         jmp ..
rcs            dap r3        ,read check sum and check subroutine
                jsp rbw
                lac sum      ,to show computed sum in ac if halt
                sas bwd
                hlt          ,check sum error
                dzm sum
ext r3         jmp ..
ext clear      szs 20        ,don't clear if ss2 up
                jmp loader
                dzm bwd
                cla
1j             sad =loader
                jmp loader
                dzm ' bwd
                idx bwd
                jmp 1j
ext =loader    loader
ext sum        ..
ext bwd        ..
ext fnl        ..
ext len        ..

```

```
ext zero-count      ..
ext loaderend      dio loader + 100
ext =dioloader      dio loader
ext p-order        ppb !
                   jmp punchoff
                   jmp end
```

```

,read binary test
,use tape with all ones and zeroes
opd      jda 170000
org 0
start    szs ' 20
          jmp test3
          lat
          dac c
          rpb
          dio temp
          lac temp
          and a
          sas a
          hlt
          lac c
          jda count
          rpb
          dio temp
          lac temp
          and b
          sas b
          hlt
          lac c
          jda count
          jmp start
count    0
          dap z
          lac count
          cma
          dac count
g         isp count
          jmp g
z         jmp
temp      0
a         770077
b         007700
c         0
t5st3     lio temp
          ppb
          lio temp2
          ppb
k         szs ' 10
          jmp start
          cli
          ppa
          jmp k
temp1     770077
temp      007700
end

```

, Address checker test program

, 7/24/61

, S. Lambert

, Low checker

org 0

start law 100 ,initial location

dap → + 4

dap check

dap temp

dzm ' → + 1

dap

idx → - 1

sas finish ,final location

jmp start + 4

lio temp ,IO contains address

check

lac

sas temp

hlt ,incorrect address

idx temp

idx check

sas trailend

jmp check - 1

zhit

szs 10 ,check one reg. continuously

jmp hit

szs 20 ,read in new tape

jmp start

read

rpb

dio temp

lac temp

dap stop

and stop

sad stop

stop

jmp

rpb

dio ' temp

jmp read

hit

lac start ,clear memory

dap → + 2

dap x + 1

dzm

idx → - 1

sas last

jmp → - 3

lat

and stp

sza '

jmp zhit

lat

dap → + 1

x

dap

lac ,check all reg. to find the

sza ,location of test word + address

jmp → + 6

idx → - 3

lio → - 4

,IO has address of reg. being checked

sas trailend

jmp x + 1

jmp zhit

```
      sas ' x
      hlt
      jmp zhit
temp  0
finish dap 7777
stp   7700
last  dzm 7777
trailend lac 7777
jmp start end .
```

...typewriter control, tyc, converted to Decal July 26, 1961,sjs

...

tyc'.a.. dac ac

```
start:   dio io
         jsp get1
         law mm      ...mm is location 2 preceding tables
         dap mg
mloop:   idx mg
         idx mg
         sad mgmax
         jmp start
         lac'mg
         sas ss1
         jmp mloop
         idx mg
         lac'mg
         dac exec
m3:      lac ac
         lio io
exec.    loc
         jmp nosk
         dac ac
         dio io
         lio cs
         tyo'
         jmp e3
nosk:    dac ac
         dio io
         jmp e3
a2:      jsp getn    ...handles a
fp4:     dio t1
         jmp start
c2:      jsp getn    ...handles c
         dio't1
         jsp step
         jmp c2      ...handles d
d2:      jsp get1
         dio'mgmax
         jsp getn
         idx mgmax
         dio'mgmax
         idx mgmax
         jmp start
emore:   jsp tcr
e2:      jsp getn    ...handles e
         dio exec
         jmp m3
e3:      szs 1
         jmp emore
         jmp start
f2:      cla        ...handles f
         dac fcount
         jsp getn
         dio beg1
         jsp getn
         dio value
begin:   lac'beg1
         xor value
         and mask
         sza
         jmp test
         idx fcount
         szs 1
         jmp test
```

```

d5:      jsp tcr
        lio beg1
        jsp type
        szs 2
        jmp test
        jsp ttab
        lio beg1
        jsp type
test:    lac beg1
        sad max
        jmp prcount
        idx beg1
        jmp begin
prcount: lio p
        tyo'
        lio fcount
        jsp type
        jmp start
        idx t1
        jmp ep4e.      lio't1
        jsp type
        jsp step
ep3:     jmp e
ep4:     jsp tcr
        lio t1
        jsp type
        jsp ttab
        jmp start
step:    dap term1
        szs 1
        jmp →+2
        jmp start
        jsp tcr
        idx t1
        lio t1
        jsp type
        jsp ttab
term1:   jmp ep3
tcr:     lio csp3
        dap →+2
        tyo'
        jmp tcr-1
ttab:    lio csp4
        jmp tcr+1
getn':   dap get1m1
        dzm mgp6
getnp2:  jsp get1
        lac ss1
        sad csm1
        jmp →+6
        and csp2
        lor mgp6
        ral 3
        dac mgp6
        jmp getnp2
        lio mgp6
        rlr 3
get1m1:  jmp nosk+4
get1:    dap typem1
        lac ac
        lio io
        szf 1
        jmp →+2
        jmp →-2
        cli clf 1
        tyi

```

```

        dio ss1
        lac ss1
        sad cret
        jmp start
typem1: jmp start+1
type':  dap →+7
        dio mgp7
        lac mgp7
        sza
        jmp →+4
        lio acm1
        tyo'
sm4:    jmp tcr-2
        cli
        dio mmm1
        dio mmp1
s:       lac mmm2
        and acm2
        sza
        jmp r
        lac mmp1
        sza
        jmp rm3
sp7:     lac mmm2
        ral 3
        dac mmm2
        idx mmm1
        sas csp1
        jmp s
        jmp sm4
rm3:     lio acm1
        tyo'
        jmp sp7
r:       lac mgp7
        cli
        rcl 3
        dac mgp7
        tyo'
        dio mmp1
        jmp r-7
csm1:    loc
cs:      loc 22
csp1:    loc 6
csp2:    loc 7
cret:csp3: loc 77
csp4:    loc 36
acm2:    law 0
acm1:    loc 20
ac:      jmp a
io:      jmp a
mask:    loc 7777
t1:      loc t1
ss1:     loc 23
mg:      loc jmpe
mgp1:mgmax: loc b2
fcount:  loc 2
beg1:    loc 7777
value:   loc jspgetn
max:     loc 7777
mgp6:    loc 12010
mmm2:mgp7: loc
mmm1:    loc 6
mm:      loc
qp2:mmp1: loc 1
        loc 61
        jmp a2

```

```

p:      loc 63
        jmp a2+3
        loc 64
        jmp d2
        loc 65
        jmp e3-3
        loc 66
        jmp e3+3
        loc 71
        jmp e-2
        loc 23
jmpe:   jmp e
        loc 26
        jmp e+4
        loc 27
        jmp t
        loc 30
        jmp tp7
        loc 31
        jmp u
b2:     loc 41
        ral 3
        loc 41
        ral 3
        loc 42
        ral 3
        loc 43
        rar 3
v:      loc a
        loc d5
        loc 4201
        loc d5
        loc 4300
        jmp t
        loc 30
        jmp tp7
t:      jsp getn
        dio v
        jsp getn
        dio v+1
jspgetn: jsp getn
        dio v+2
        jmp start
tp7:    lio v
        dio v+3
        lio v+2
        dio v+4
        lio v+3
        dio v+4
        lac v+3
        sad v+1
        jmp start
        idx v+3
        idx v+4
        jmp --7
u:      lio v
        dio v+3
        lio v+2
        dio v+4
        lac v+3
        sas v+4
        jmp ++7
        lac v+3
        sad v+1
        jmp start
        idx v+3

```

```
idx v+4  
jmp →-10  
jsp tcr  
lio v+3  
jsp type  
jsp ttab  
lio'v+3  
jsp type  
jmp →-14
```

fin

/codeword display

```
cwd,      0
          dap cwX
          lac i cWd
          dac  $\overline{cW}$ 
          idx cWd
          xct  $\frac{1}{2}$  cWx
          dac  $\overline{x}$ 
          idx cWx
          xct 1 cWx
          lio i cWd
          spi
          sub  $\overline{2}dl$ 
          dac  $\overline{y}$ 
          idx cWx
          setup  $\overline{ctr}$ , 22
          setup  $\overline{cty}$ , 7

d,         setup  $\overline{ctx}$ , 5

c,         lio  $\overline{cW}$ 
          ril  $\frac{1}{2}s$ 
          dio  $\overline{cW}$ 
          spi
          jmp plt

a,         isp  $\overline{ctr}$ 
          jmp b
          lac i cWd
          dac cW
          setup  $\overline{ctr}$ , 22

b,         isp ctx
          jmp inx
          isp cty
          jmp iny

cWx,       jmp .

inx,       lac x
          add del
          dac x
          jmp c

iny,       lac x
          sub  $\overline{4}dl$ 
          dac x
          lac y
          add del
          dac y
          jmp d

plt,       lac x
          lio y
          dpy-i
          jmp a
```

law cW1
ida cWd
lac lx
lac ly

```
siz,del, 2000
dap sex
lac del
sal 1s
dac 2dl
sal 1s
dac 4dl
sex, jmp .
```

```
variables
constants
```

```
start
```

/alphabetic codeword tables

lwr,	374200	000000	/space, printing
	561020	010604	/1
	774040	005056	/2
	564204	203056	/3
	420413	345122	/4
	164204	175037	/5
	564307	241056	/6
	604040	004077	/7
	564305	243056	/8
	564205	343056	/9
	0	0	
	0	0	
	0	0	
	0	0	
	0	0	
	0	0	
	0	0	
	144512	245114	/0
	602020	010101	//
	160231	016000	/s
	421220	010744	/t
	072245	122000	/u
	042506	142000	/v
	125326	142000	/w
	212421	242000	/x
	060235	522451	/y
	372020	276000	/z
	0	0	
	440430	600000	/,
	0	0	
	0	0	
	001170	174200	/tab, printing
	0	0	
	400000	214000	/.
	042410	614002	/j
	112461	222410	/k
	070410	204106	/l
	655326	164000	/m
	512245	124000	/n
	062245	114000	/o
	102071	522456	/p
	030472	645116	/q
	502041	124000	/r
	0	0	
	0	0	
	000174	000000	/-
	101010	204210	/)
	000000	000037	/^
	021041	020202	/(\
	0	0	
	072234	114000	/a
	162245	134410	/b
	072041	016000	/c
	072245	116041	/d
	072075	114000	/e
	102161	020446	/f
	060235	522446	/g

PATTERN :

	13	14	15	16	17
WORD 2	8	9	10	11	12
	3	4	5	6	7
	16	17	0	1	2
WORD 1	11	12	13	14	15
	6	7	8	9	10
	1	2	3	4	5

WORD 1, BIT 0: DEPRESS 2.

	112245	134410	/h
	561020	030004	/i
	042555	224512	/l.c., printing
	061400	000000	/.
	122451	266504	/u.c., printing
	042175	010000	/bksp, printing
	0	0	
	306020	204600	/c.r., printing
upr,	374200	000000	/space, printing
	000000	023562	/"
	000000	004304	/1
	000000	005250	/~
	760207	200000	/D
	042452	142000	/V
	214251	210000	/^
	010421	010101	/<
	101010	104210	/>
	441020	052704	/↑
	0	0	
	0	0	
	0	0	
	0	0	
	0	0	
	0	0	
	040574	210000	/→
	040020	243056	/?
	564205	241056	/S
	441020	010237	/T
	164306	143061	/U
	042506	143061	/V
	125326	143061	/W
	612420	010521	/X
	441020	010521	/Y
	774040	004077	/Z
	0	0	
	407603	300000	/=
	0	0	
	0	0	
	050774	312000	/tab, printing
	0	0	
	007600	400000	/
	144410	204107	/J
	214523	051121	/K
	374102	041020	/L
	214306	153561	/M
	214306	353461	/N
	164306	143056	/O
	604103	243076	/P
	154526	143056	/Q
	614523	243076	/R

0	0	
0	0	
441174	010000	/+
060410	204106	/]
441020	010204	/]
461020	010206	/[
0	0	
214376	143056	/A
764307	243076	/B
164302	041056	/C
364306	143076	/D
774103	041037	/E
604103	041037	/F
164336	041056	/G
614307	343061	/H
561020	010216	/I
042555	224512	/l.c., printing
404250	025040	/x
122451	266504	/u.c., printing
042175	010000	/bksp, printing
0	0	
306020	204600	/c.r., printing

start

/codeword digit display

```
cw $\overline{d}$ ,      0
            dap cw $\overline{x}$ 
            lac cw $\overline{d}$ 
            sad (20
            law 12
            sad (54
            law 13
            ral 1s
            add (1w $\overline{r}$ 
            dac cw $\overline{d}$ 

cwe,        lac  $\overline{i}$  cw $\overline{d}$ 
            dac  $\overline{c}w$ 
            idx cw $\overline{d}$ 
            lac  $\overline{y}0$ 
            dac  $\overline{y}$ 
            setup  $\overline{c}tr$ , 22
            setup  $\overline{c}ty$ , 7

d,          setup  $\overline{c}tx$ , 5

c,          lio  $\overline{c}w$ 
            ril 1s
            dio  $\overline{c}w$ 
            spi
            jmp plt

a,          isp  $\overline{c}tr$ 
            jmp b
            lac  $\overline{i}$  cw $\overline{d}$ 
            dac  $\overline{c}w$ 
            setup  $\overline{c}tr$ , 22

b,          isp  $\overline{c}tx$ 
            jmp inx
            isp  $\overline{c}ty$ 
            jmp iny
            lac x
            add 2d1
            dac x

cw $\overline{x}$ ,      jmp .

inx,        lac x
            add del
            dac x
            jmp c

iny,        lac x
            sub 4d1
            dac x
            lac y
            add del
            dac y
            jmp d
```

```

plt,      lac x
          lio y
          dpy-i
          jmp a

siz,del,  2000
          dap sex
          lac del
          sal 1s
          dac 2dl
          sal 1s
          dac 4dl
sex,      jmp .

2dl,      4000
4dl,      10000

so,x,     0      /set origin
          dap sox
          dio y0
sox,      jmp .

y0,       0

```

variables
constants

/alphabetic codeword tables

lwr,	000000	000000	/space
	561020	010604	/1
	774040	005056	/2
	564204	203056	/3
	420413	345122	/4
	164204	175037	/5
	564307	241056	/6
	604040	004077	/7
	564305	243056	8
	564205	343056	/9
	144512	45114	/0
	000174	000000	/-

start