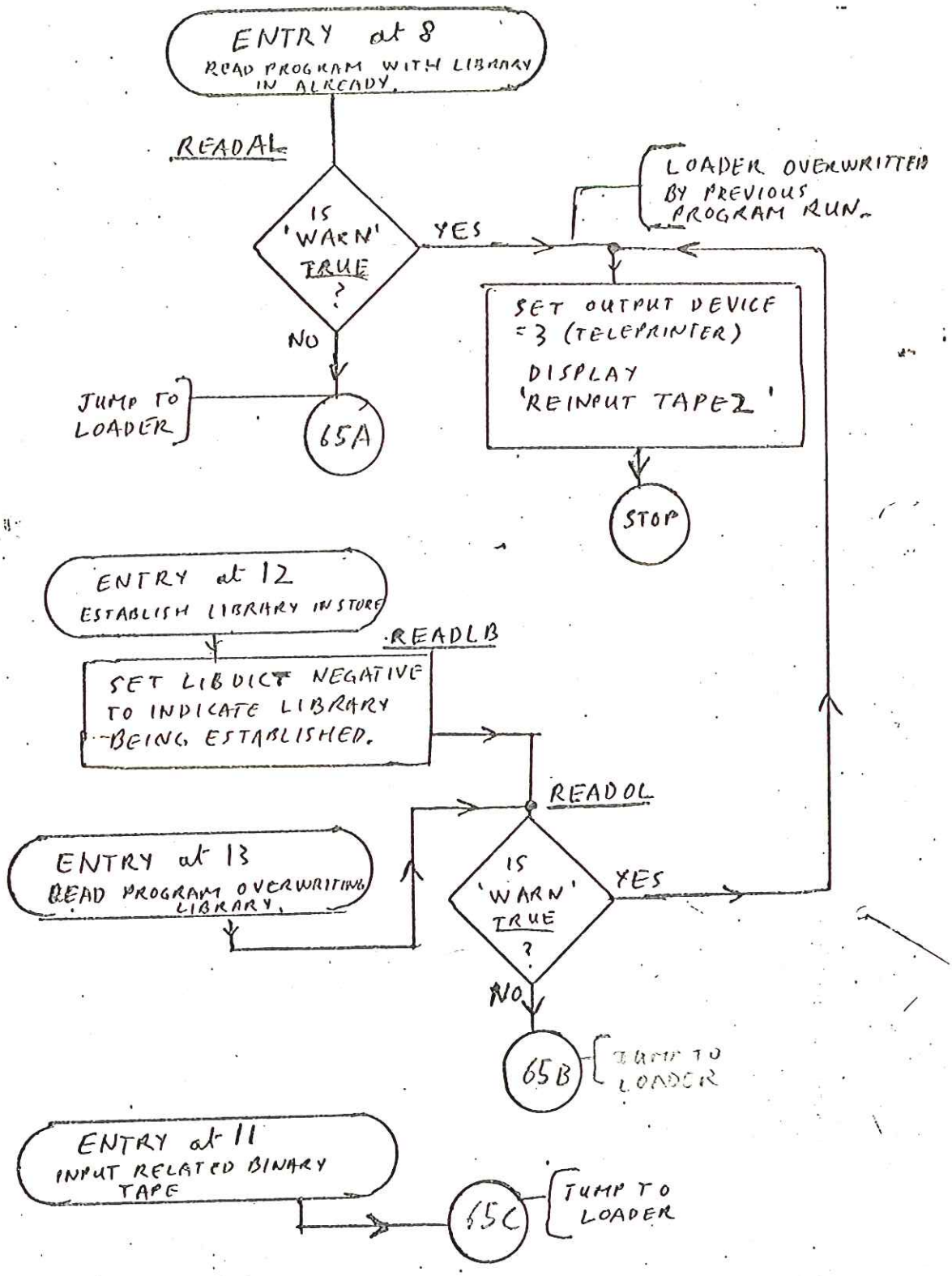
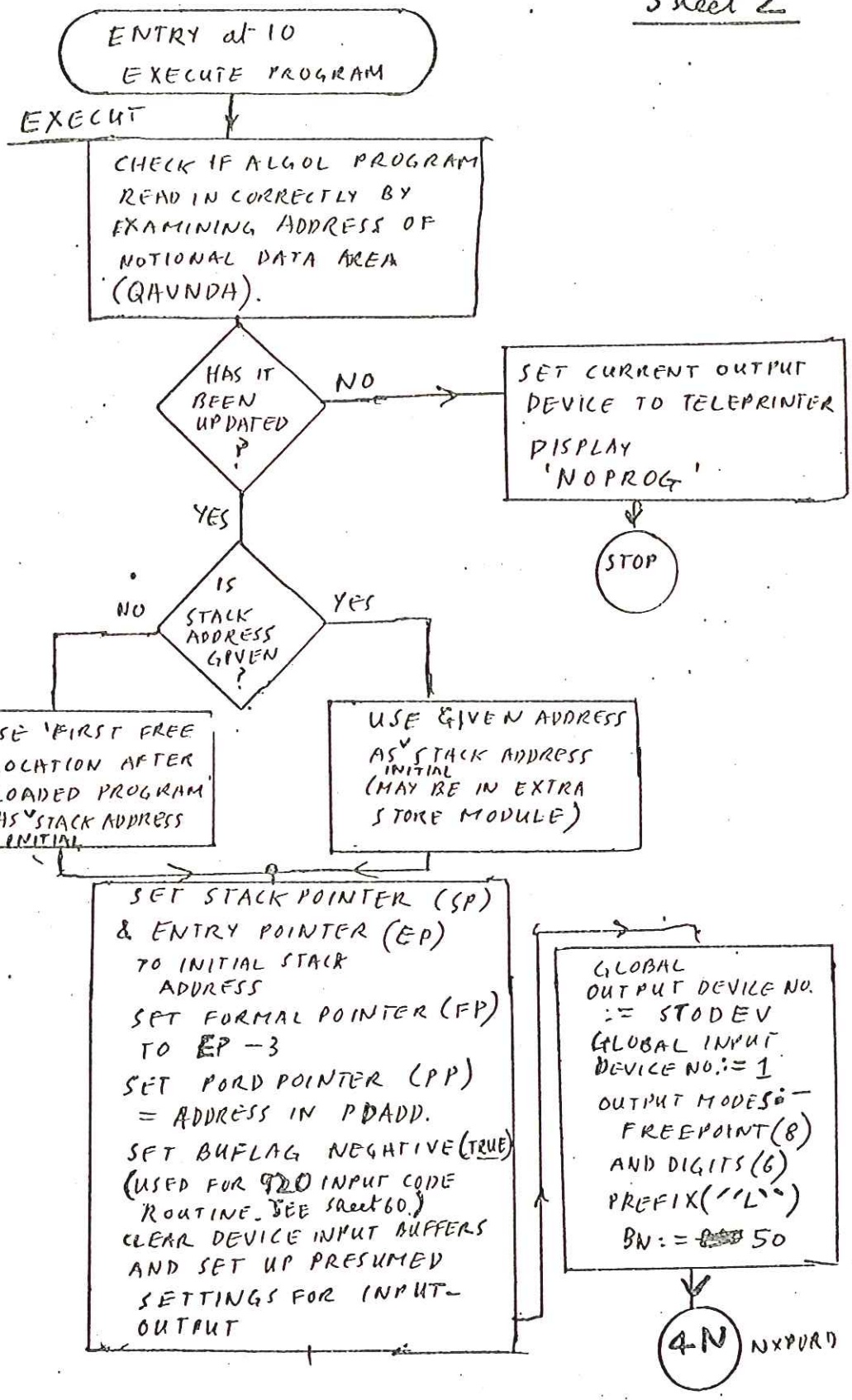
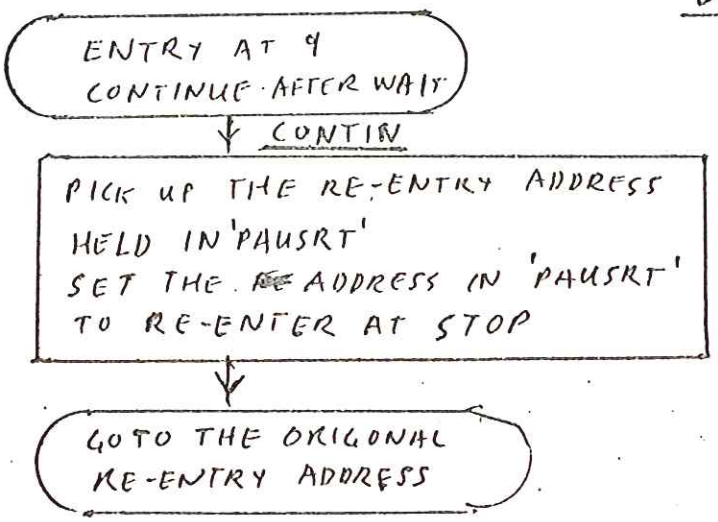
# 903 ALGOL INTERPRETER

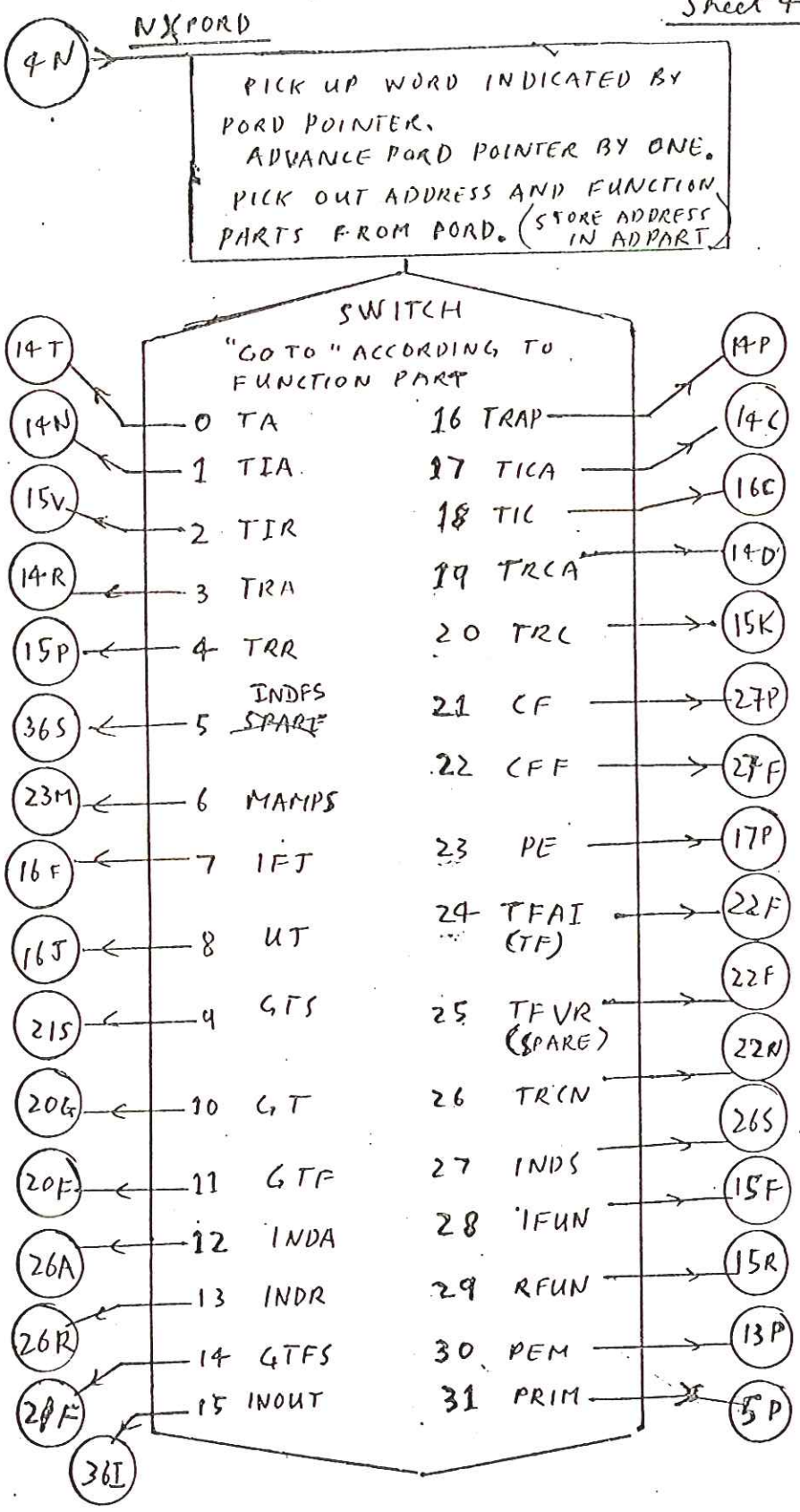


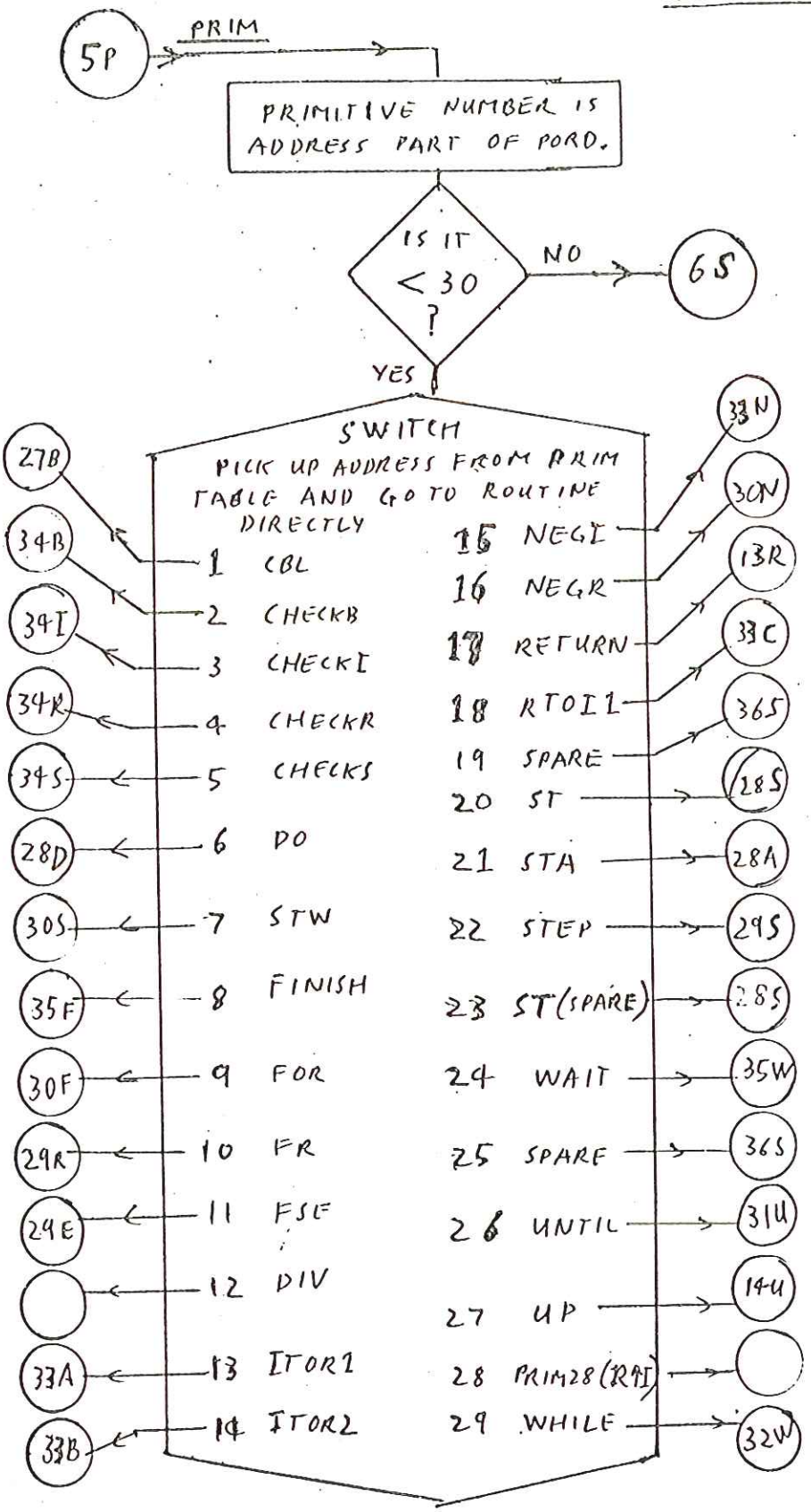


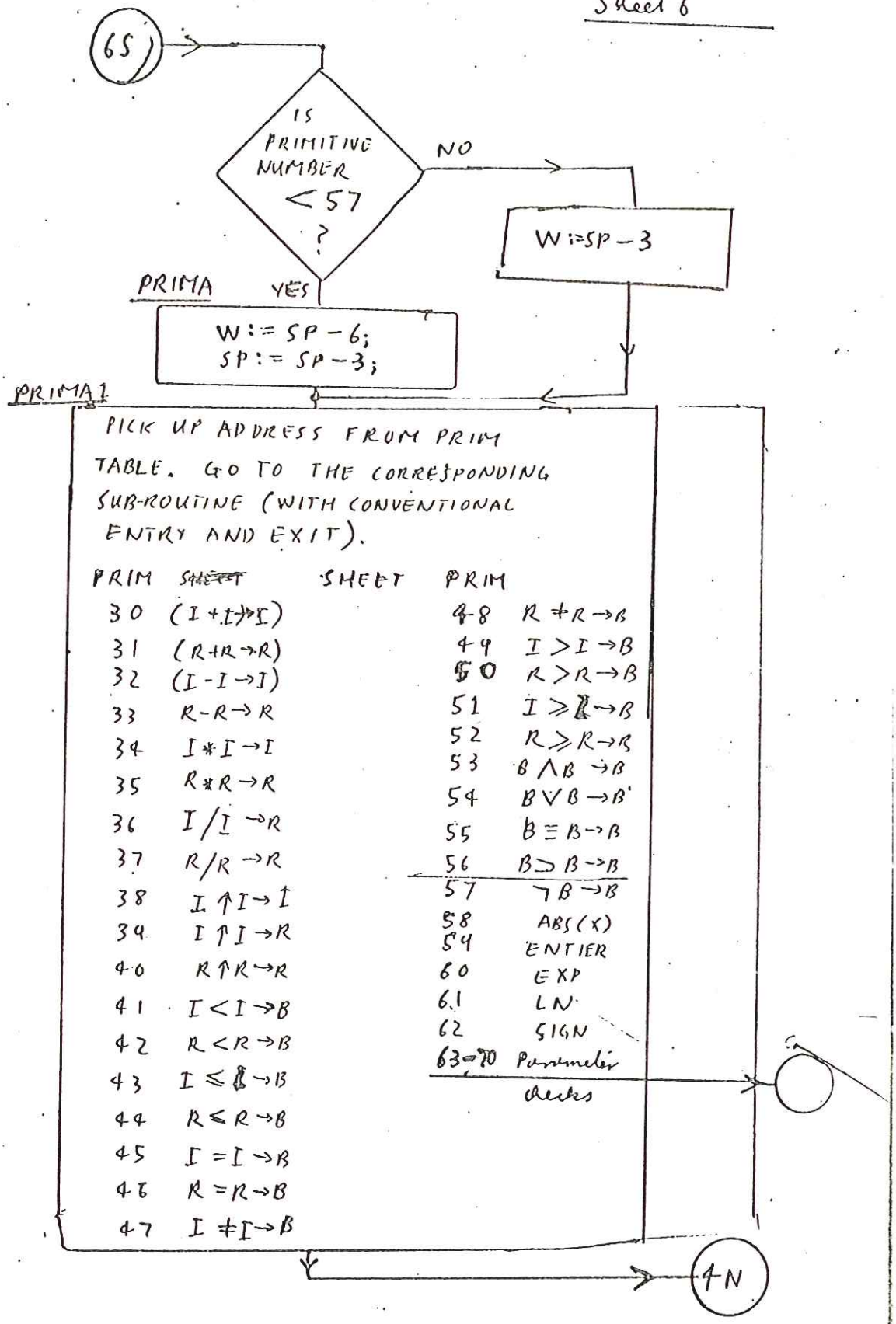

ENTRY at 10
EXECUTE PROGRAM
EXECUT
CHECK IF ALGOL PROGRAM READ IN CORRECTLY BY EXAMINING ADDRESS OF NOTIONAL DATA AREA (QAUNDA).
HAS IT BEEN UPDATED ?
NO
SET CURRENT OUTPUT DEVICE TO TELEPRINTER
DISPLAY 'NOPROG'
STOP
YES
IS STACK ADDRESS GIVEN ?
NO
YES
USE 'FIRST FREE LOCATION AFTER LOADED PROGRAM' AS STACK ADDRESS INITIAL
USE GIVEN ADDRESS AS STACK ADDRESS INITIAL (MAY BE IN EXTRA STORE MODULE)
SET STACK POINTER (SP) & ENTRY POINTER (EP) TO INITIAL STACK ADDRESS
SET FORMAL POINTER (FP) TO EP -3
SET PORD POINTER (PP) = ADDRESS IN PDADD.
SET BUFLAG NEGATIVE (TRUE) (USED FOR 920 INPUT CODE ROUTINE. SEE Sheet 60.)
CLEAR DEVICE INPUT BUFFERS AND SET UP PRESUMED SETTINGS FOR INPUT-OUTPUT
GLOBAL OUTPUT DEVICE NO. := STODEV
GLOBAL INPUT DEVICE NO.:= 1
OUTPUT MODES:-
FREEPOINT(8)
AND DIGITS(6)
PREFIX("L")
BN:= 50
4-N
NXPORD



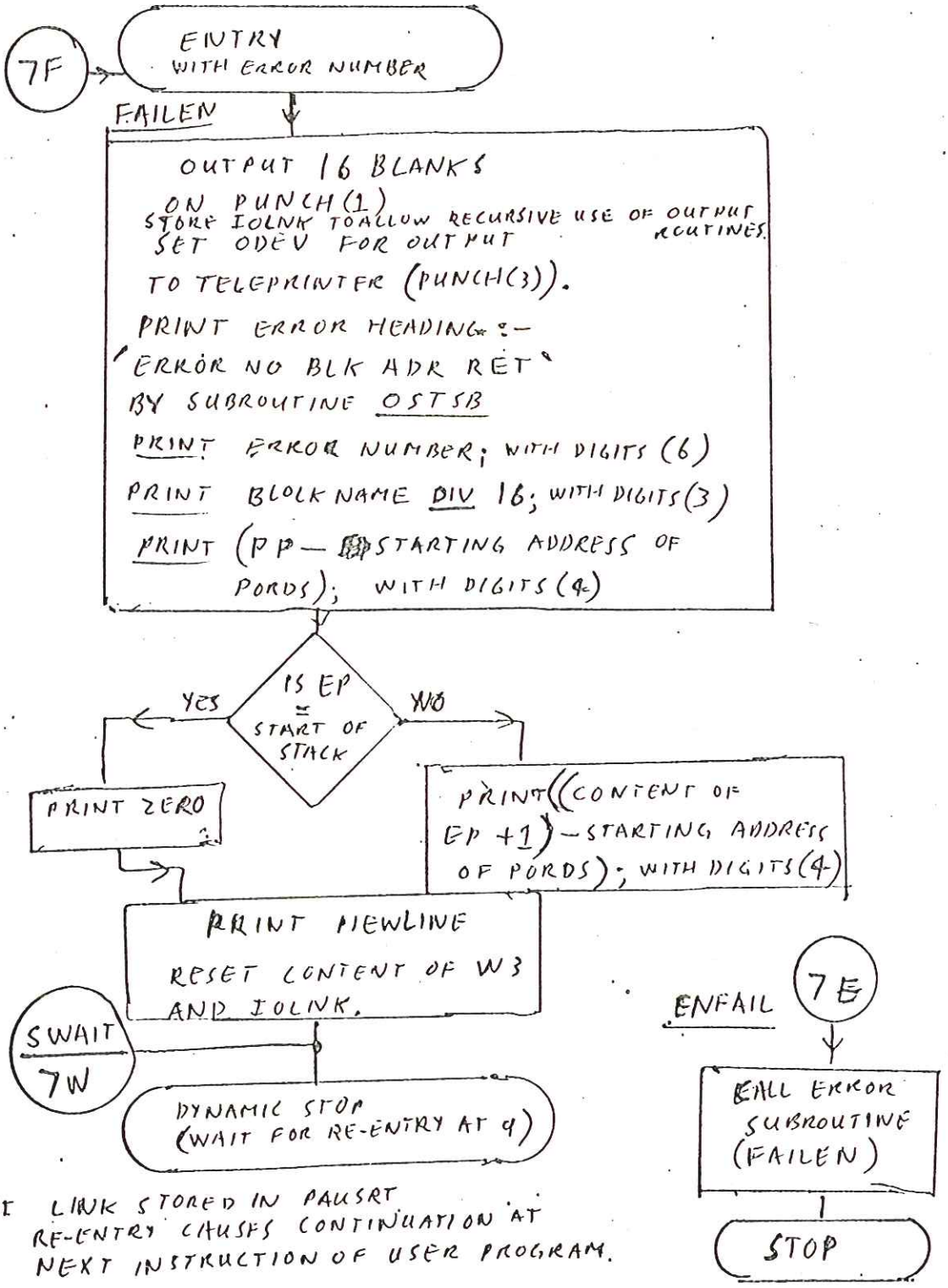

Room for further extensions, e.g. level change program and level 3 Skimulus


4N
NXPORD
PICK UP WORD INDICATED BY PORD POINTER.
ADVANCE PORD POINTER BY ONE.
PICK OUT ADDRESS AND FUNCTION PARTS FROM PORD. (STORE ADDRESS IN ADPART)
SWITCH
"GO TO" ACCORDING TO FUNCTION PART
14T
14N
15V
14R
15P
36S
23M
16F
16J
21S
20G
20F
26A
26R
28F
36I
0 TA
1 TIA
2 TIR
3 TRA
4 TRR
5 INDFS SPARE
6 MAMPS
7 IFJ
8 UT
9 GTS
10 GT
11 GTF
12 INDA
13 INDR
14 GTFS
15 INOUT
16 TRAP
17 TICA
18 TIC
19 TRCA
20 TRC
21 CF
22 CFF
23 PE
24 TFAI (TF)
25 TFVR (SPARE)
26 TRCN
27 INDS
28 IFUN
29 RFUN
30 PEM
31 PRIM
14P
14C
16C
14D'
15K
27P
27F
17P
22F
22F
22N
26S
15F
15R
13P
5P


5P
PRIM
PRIMITIVE NUMBER IS ADDRESS PART OF PORD.
IS IT < 30 ?
NO
6S
YES
SWITCH
PICK UP ADDRESS FROM PRIM TABLE AND GO TO ROUTINE DIRECTLY
1 CBL
2 CHECKB
3 CHECKI
4 CHECKR
5 CHECKS
6 DO
7 STW
8 FINISH
9 FOR
10 FR
11 FSE
12 DIV
13 ITOR1
14 ITOR2
15 NEGI
16 NEGR
17 RETURN
18 RTOI1
19 SPARE
20 ST
21 STA
22 STEP
23 ST(SPARE)
24 WAIT
25 SPARE
26 UNTIL
27 UP
28 PRIM28(RAI
29 WHILE
27B
34B
34I
34R
34S
28D
30S
35F
30F
29R
29E
33A
33B
33N
30N
13R
33C
36S
28S
28A
29S
28S
35W
36S
31U
14U
32W

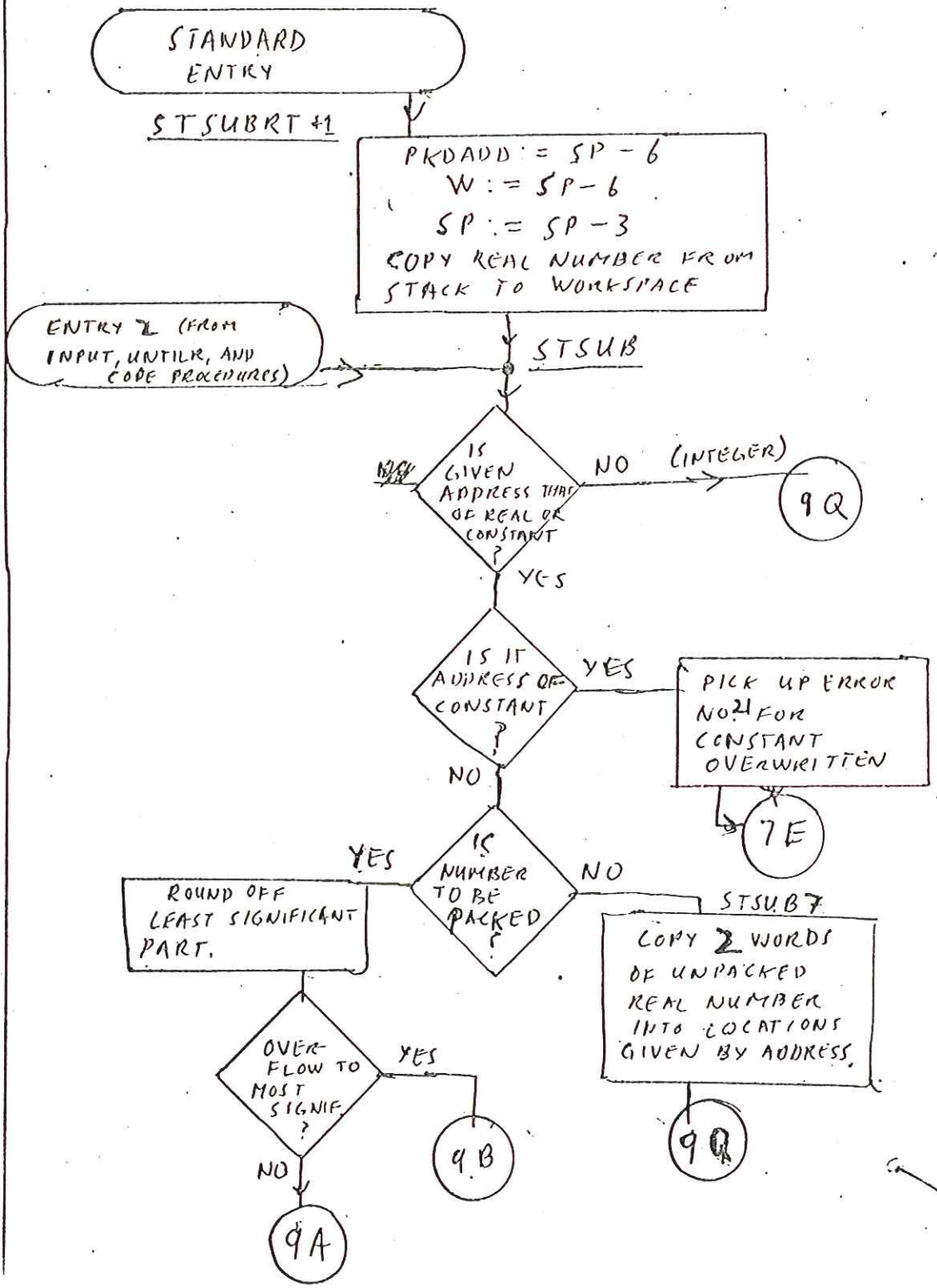

65
IS PRIMITIVE NUMBER < 57 ?
NO
W:=SP-3
PRIMA
YES
W:= SP-6;
SP:= SP-3;
PRIMA1
PICK UP ADDRESS FROM PRIM TABLE. GO TO THE CORRESPONDING SUB-ROUTINE (WITH CONVENTIONAL ENTRY AND EXIT).
PRIM SHEET
30 (I+I→I)
31 (R+R→R)
32 (I-I→I)
33 R-R→R
34 I*I→I
35 R*R→R
36 I/I→R
37 R/R→R
38 I↑I→I
39 I↑I→R
40 R↑R→R
41 I<I→B
42 R<R→B
43 I≤I→B
44 R≤R→B
45 I=I→B
46 R=R→B
47 I≠I→B
SHEET PRIM
48 R≠R→B
49 I>I→B
50 R>R→B
51 I≥I→B
52 R≥R→B
53 B∧B→B
54 B∨B→B
55 B≡B→B
56 B⊃B→B
57 ¬B→B
58 ABS(x)
59 ENTIER
60 EXP
61 LN
62 SIGN
63-70 Parameter checks
4N

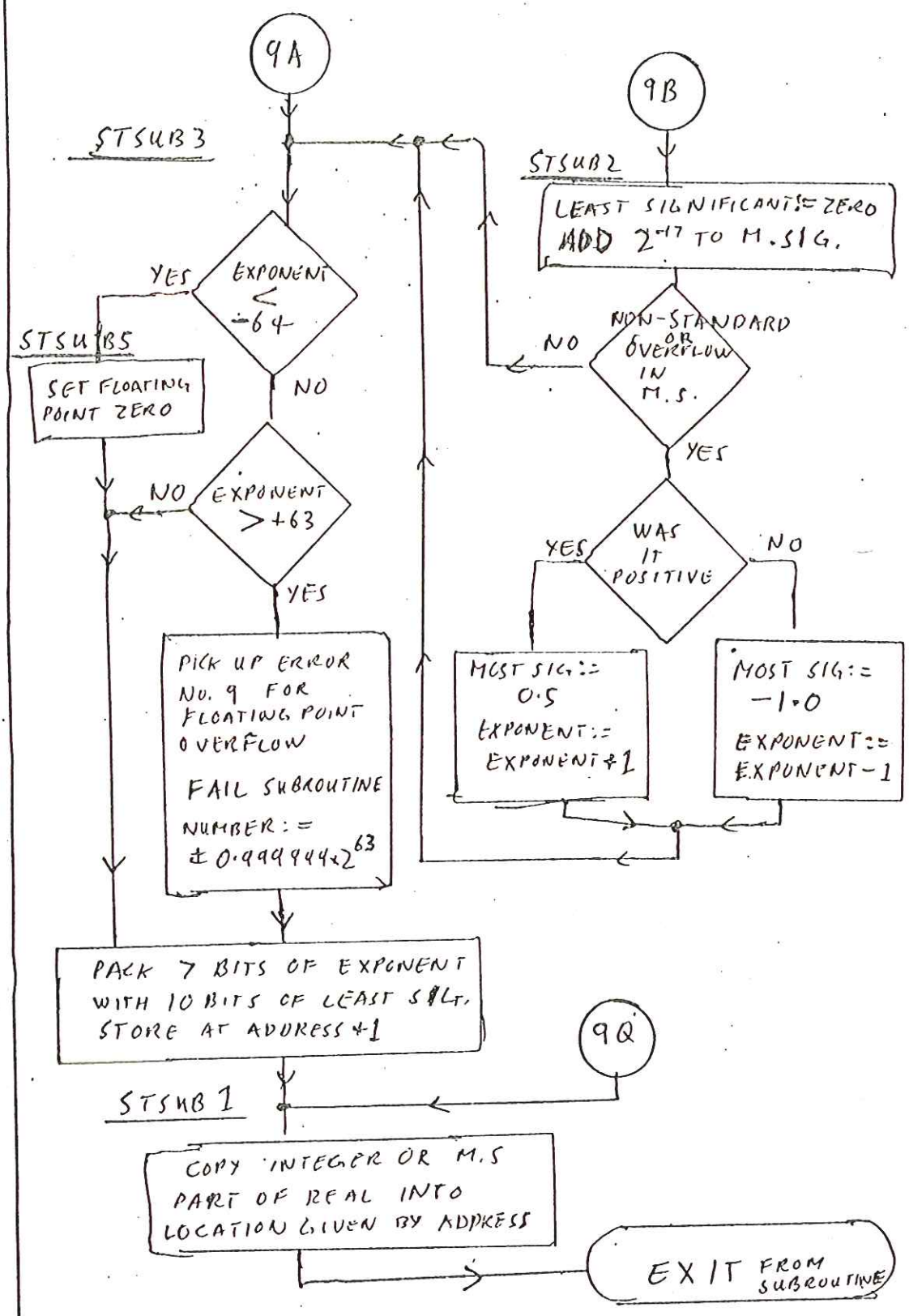
# FAIL Subroutine (FAILEN)



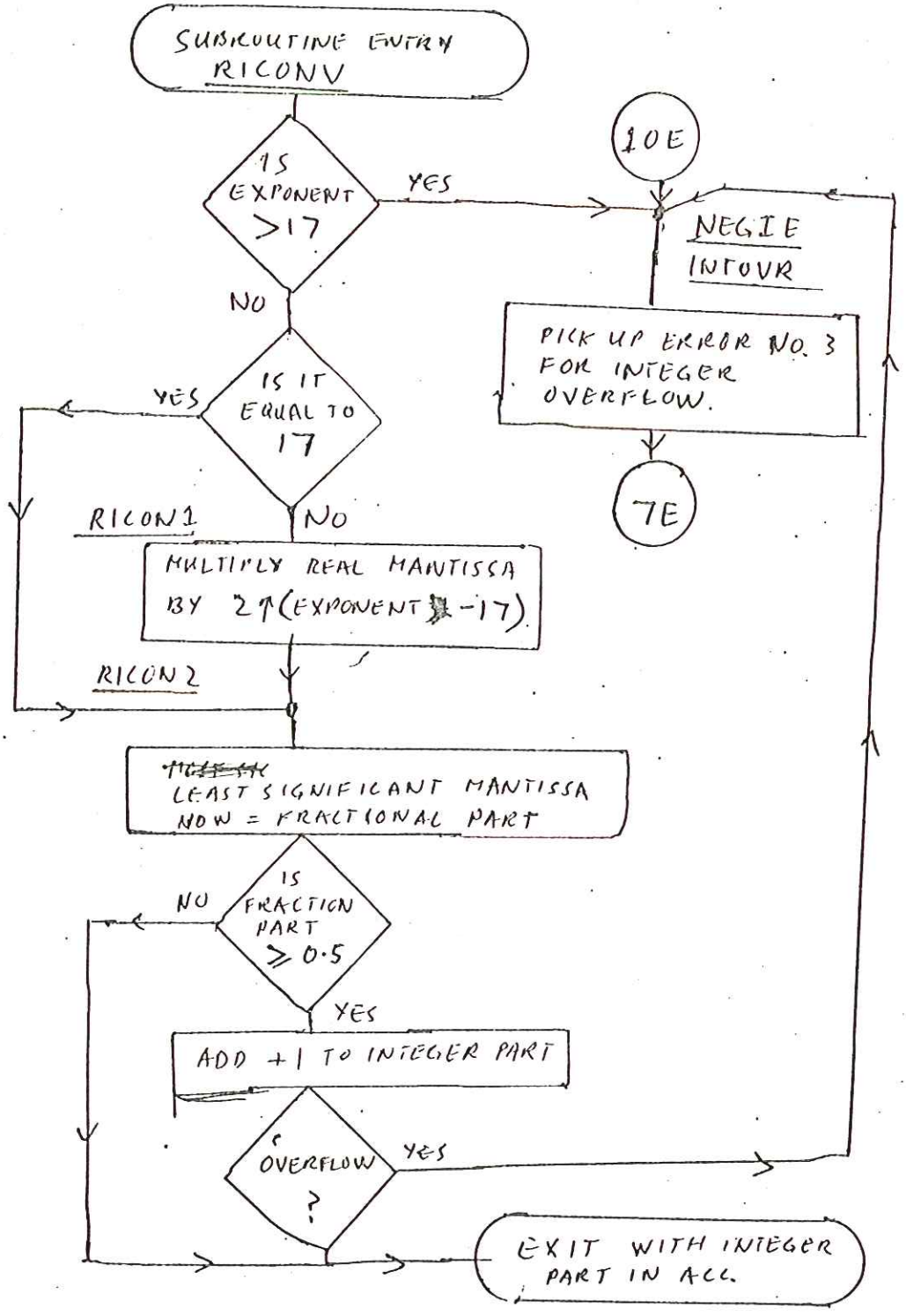

SUBRT LINK STORED IN PAUSRT
RE-ENTRY CAUSES CONTINUATION AT NEXT INSTRUCTION OF USER PROGRAM.

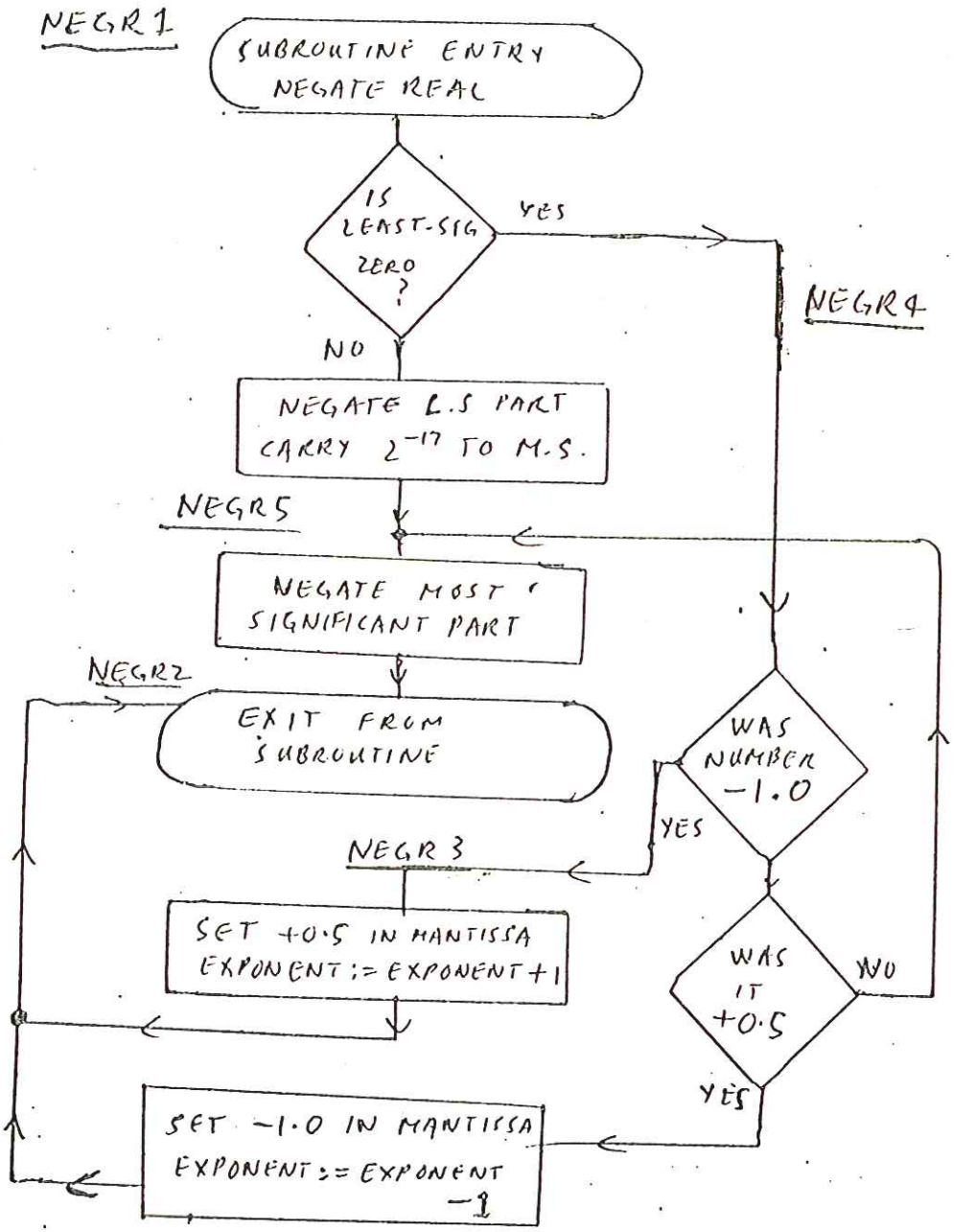
# ASSIGNE Subroutine



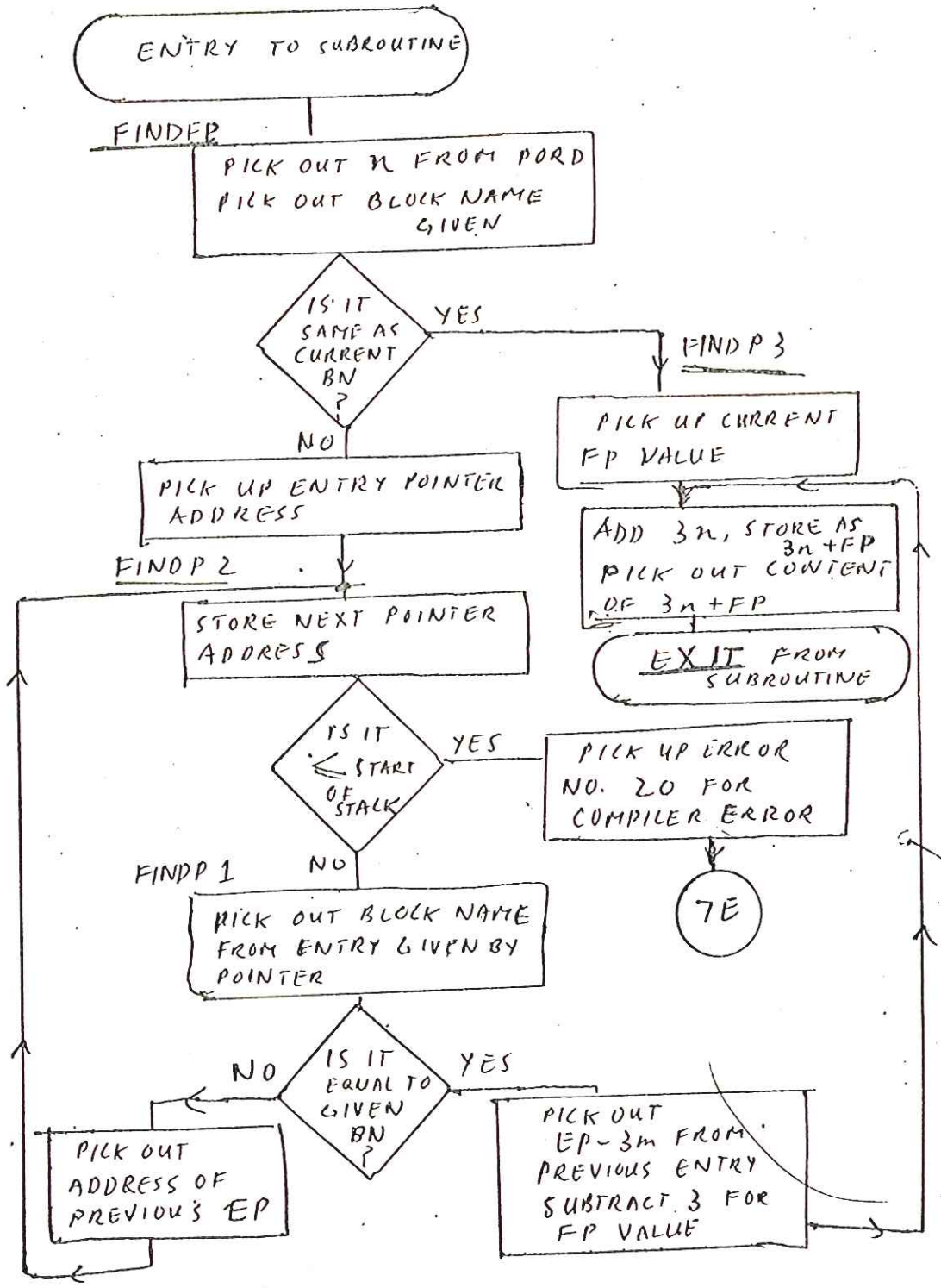

9A
9B
STSUB3
STSUB2
LEAST SIGNIFICANT := ZERO
ADD $2^{-17}$ TO M.SIG.
YES
EXPONENT < -64
NON-STANDARD OR OVERFLOW IN M.S.
NO
YES
STSUB5
SET FLOATING POINT ZERO
NO
NO
EXPONENT > +63
YES
WAS IT POSITIVE
YES
NO
PICK UP ERROR No. 9 FOR FLOATING POINT OVERFLOW
FAIL SUBROUTINE
NUMBER := $\pm 0.999999 \times 2^{63}$
MOST SIG := 0.5
EXPONENT := EXPONENT + 1
MOST SIG := -1.0
EXPONENT := EXPONENT - 1
PACK 7 BITS OF EXPONENT WITH 10 BITS OF LEAST SIG. STORE AT ADDRESS + 1
9Q
STSUB1
COPY INTEGER OR M.S PART OF REAL INTO LOCATION GIVEN BY ADDRESS
EXIT FROM SUBROUTINE

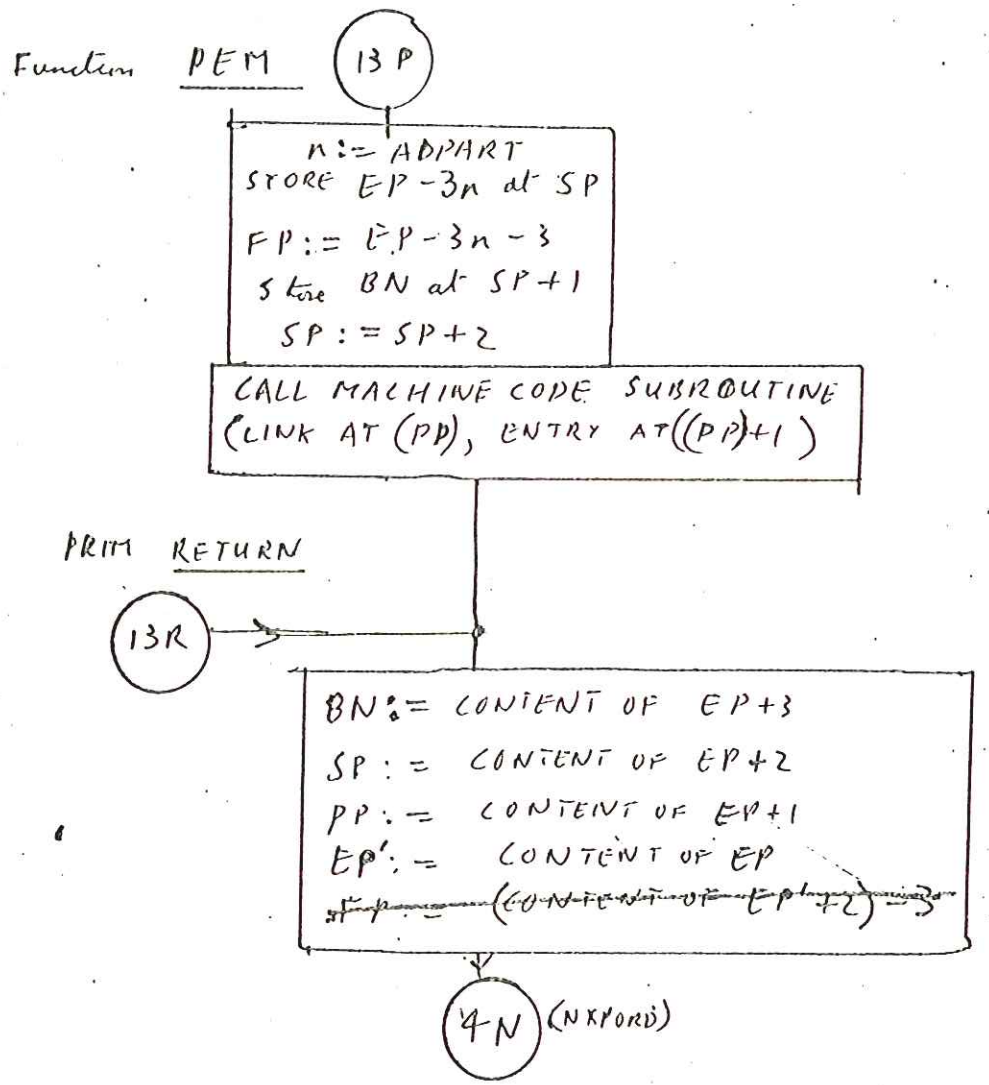
# REAL TO INTEGER SUBROUTINE

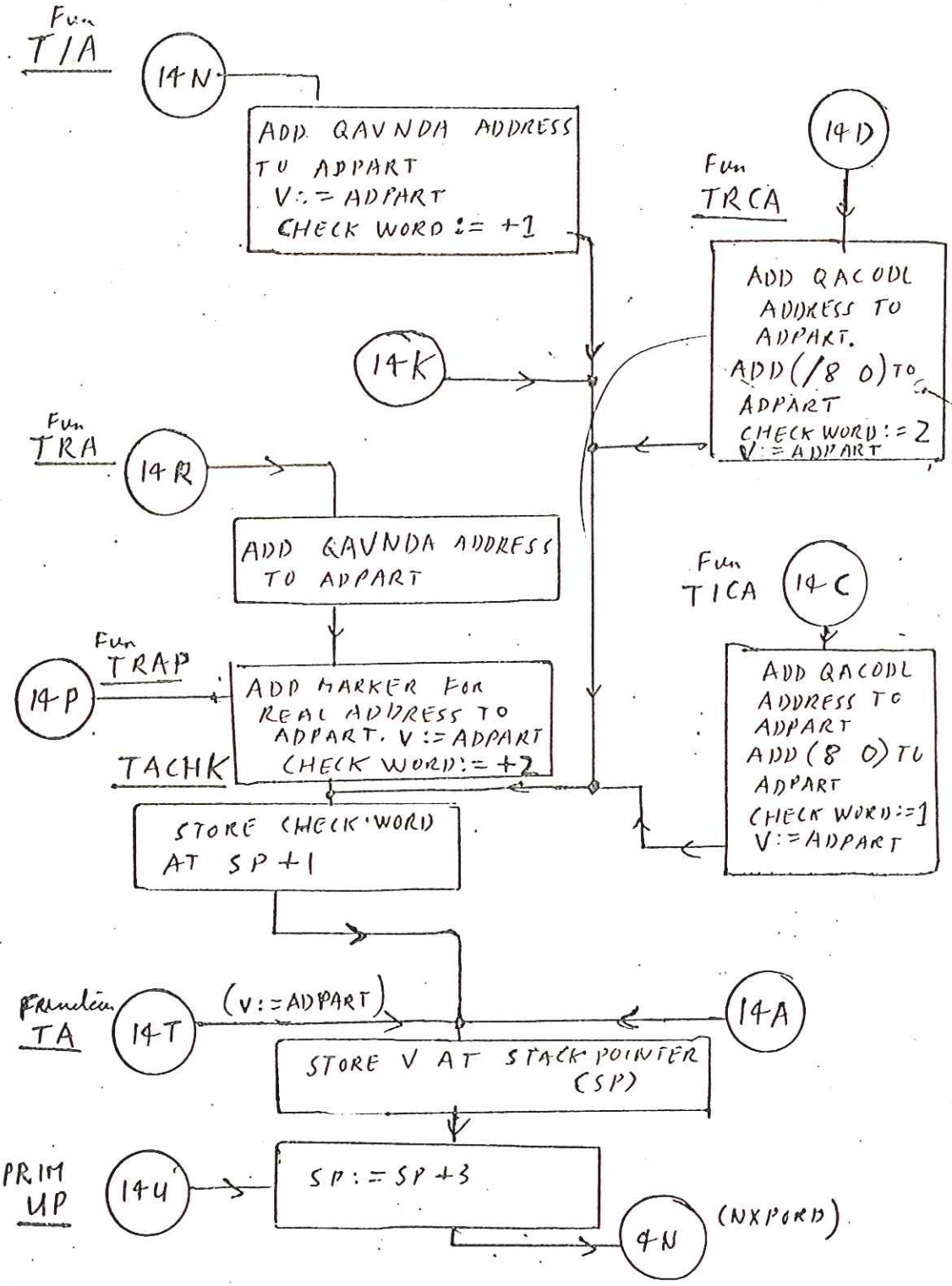


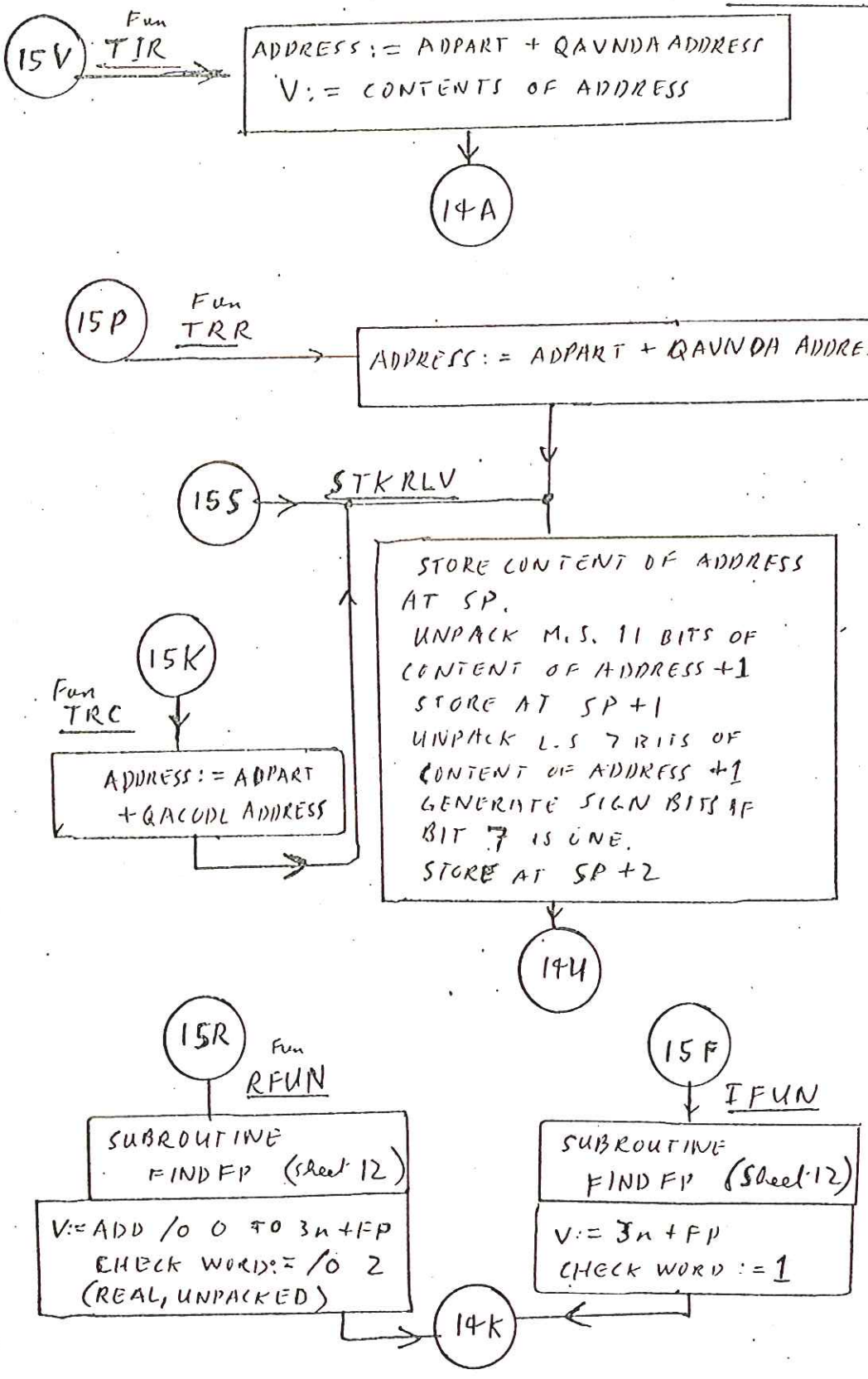

NEGR1
SUBROUTINE ENTRY
NEGATE REAL
IS LEAST-SIG ZERO ?
YES
NO
NEGR4
NEGATE L.S PART
CARRY $2^{-17}$ TO M.S.
NEGR5
NEGATE MOST SIGNIFICANT PART
NEGR2
EXIT FROM SUBROUTINE
WAS NUMBER −1.0
YES
NEGR3
SET +0.5 IN MANTISSA
EXPONENT := EXPONENT+1
WAS IT +0.5
NO
YES
SET −1.0 IN MANTISSA
EXPONENT := EXPONENT −1

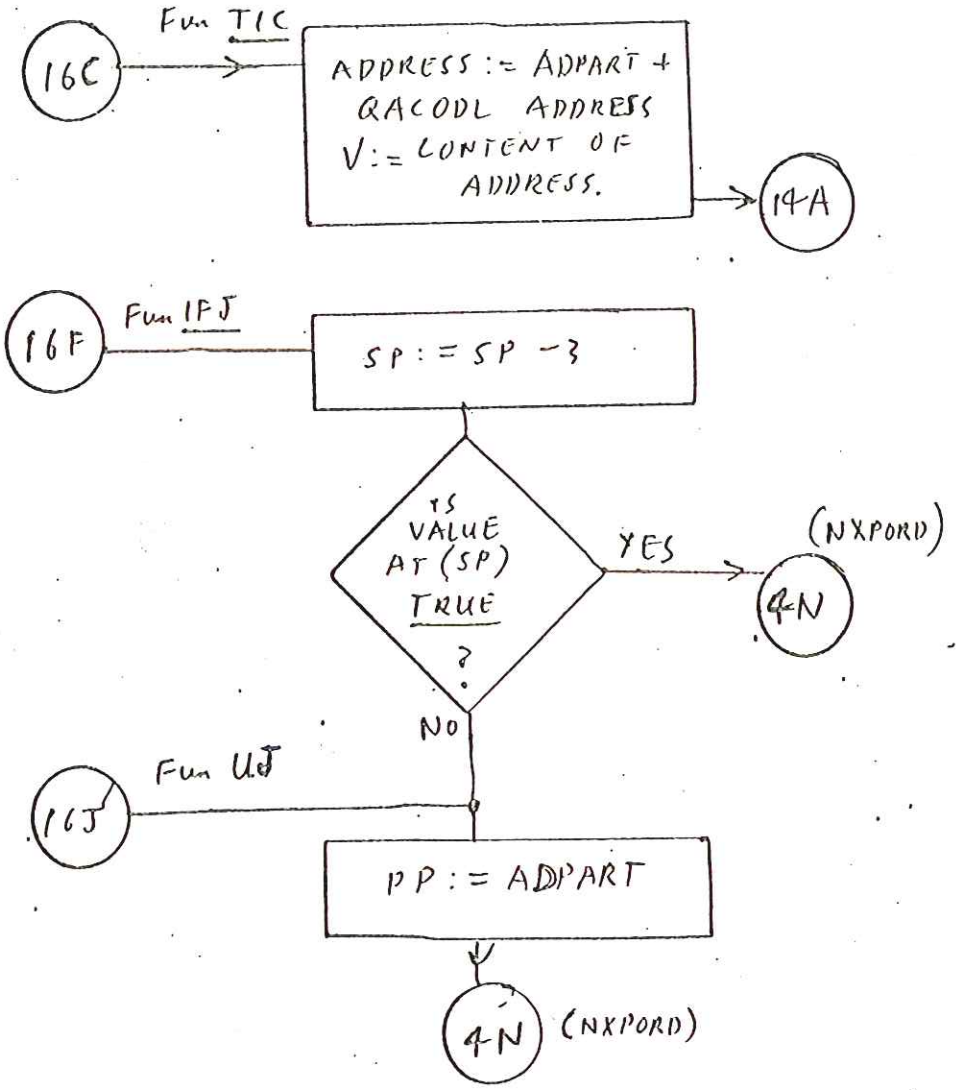
SUBROUTINE: FIND FORMAL POINTER AND PARAMETER POSITION

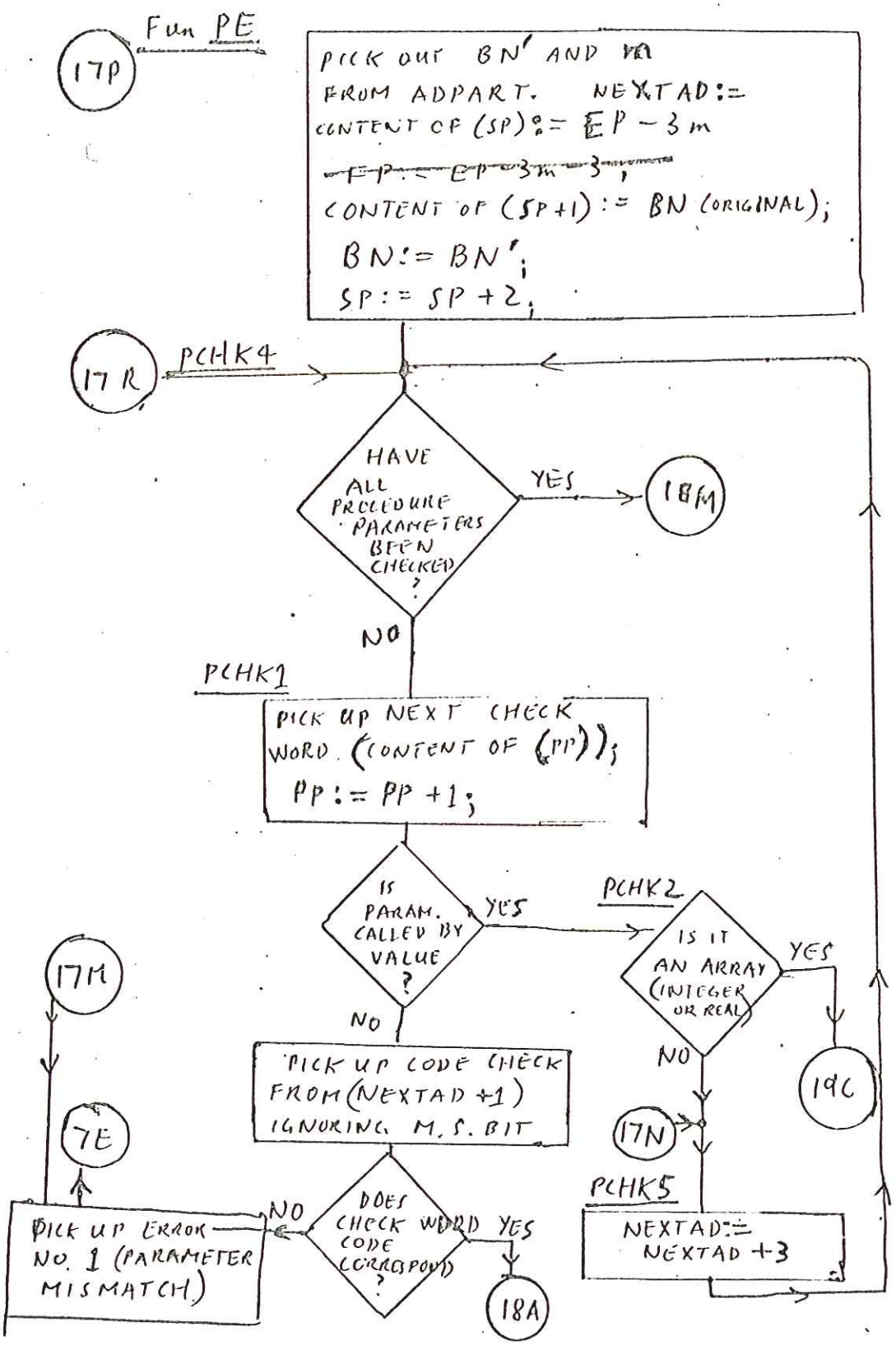


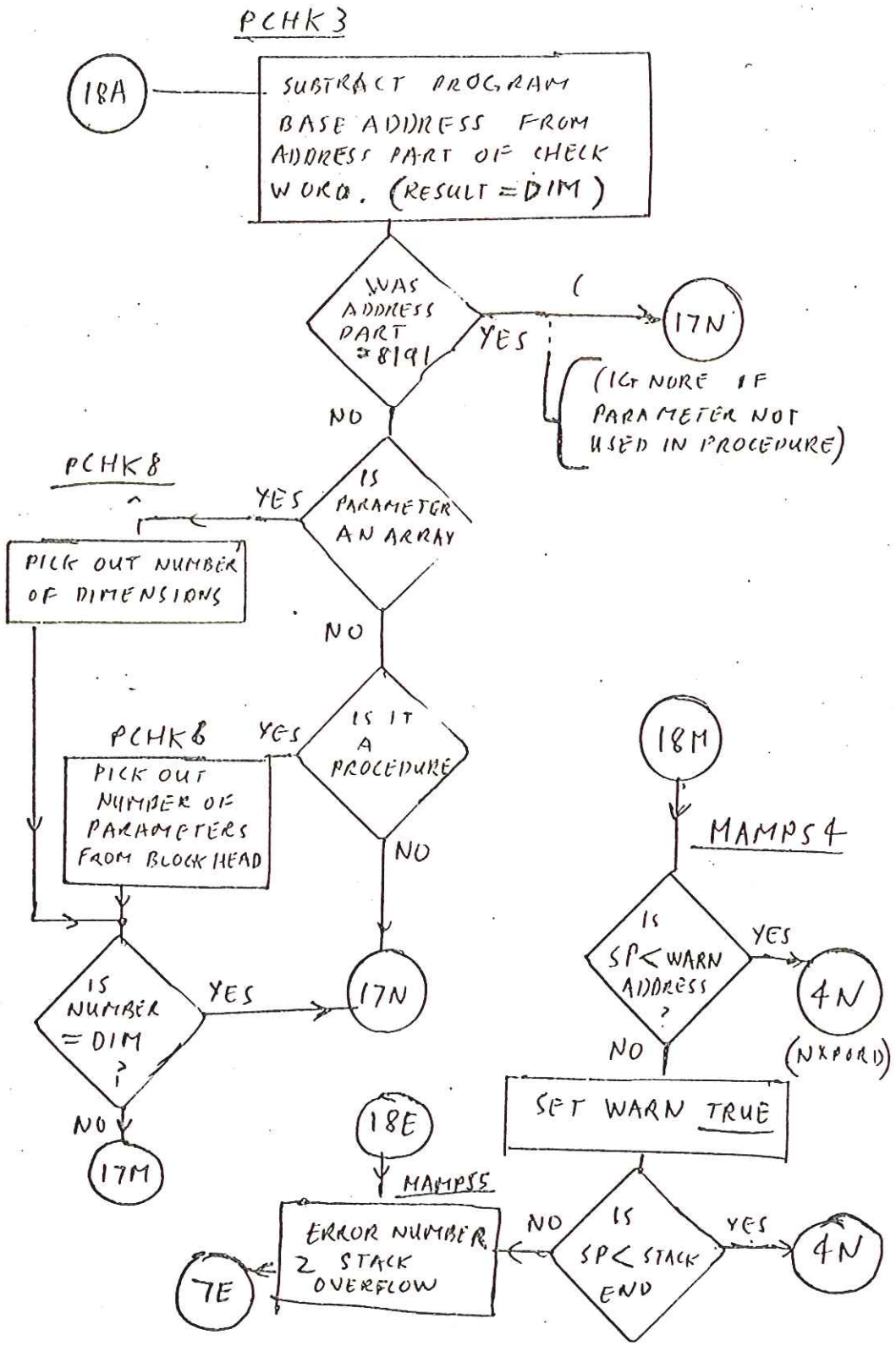

Function PEM
13P
n := ADPART
STORE EP-3n at SP
FP := EP-3n-3
Store BN at SP+1
SP := SP+2
CALL MACHINE CODE SUBROUTINE
(LINK AT (PP), ENTRY AT ((PP)+1))
PRIM RETURN
13R
BN := CONTENT OF EP+3
SP := CONTENT OF EP+2
PP := CONTENT OF EP+1
EP := CONTENT OF EP
4N
(NXPORD)

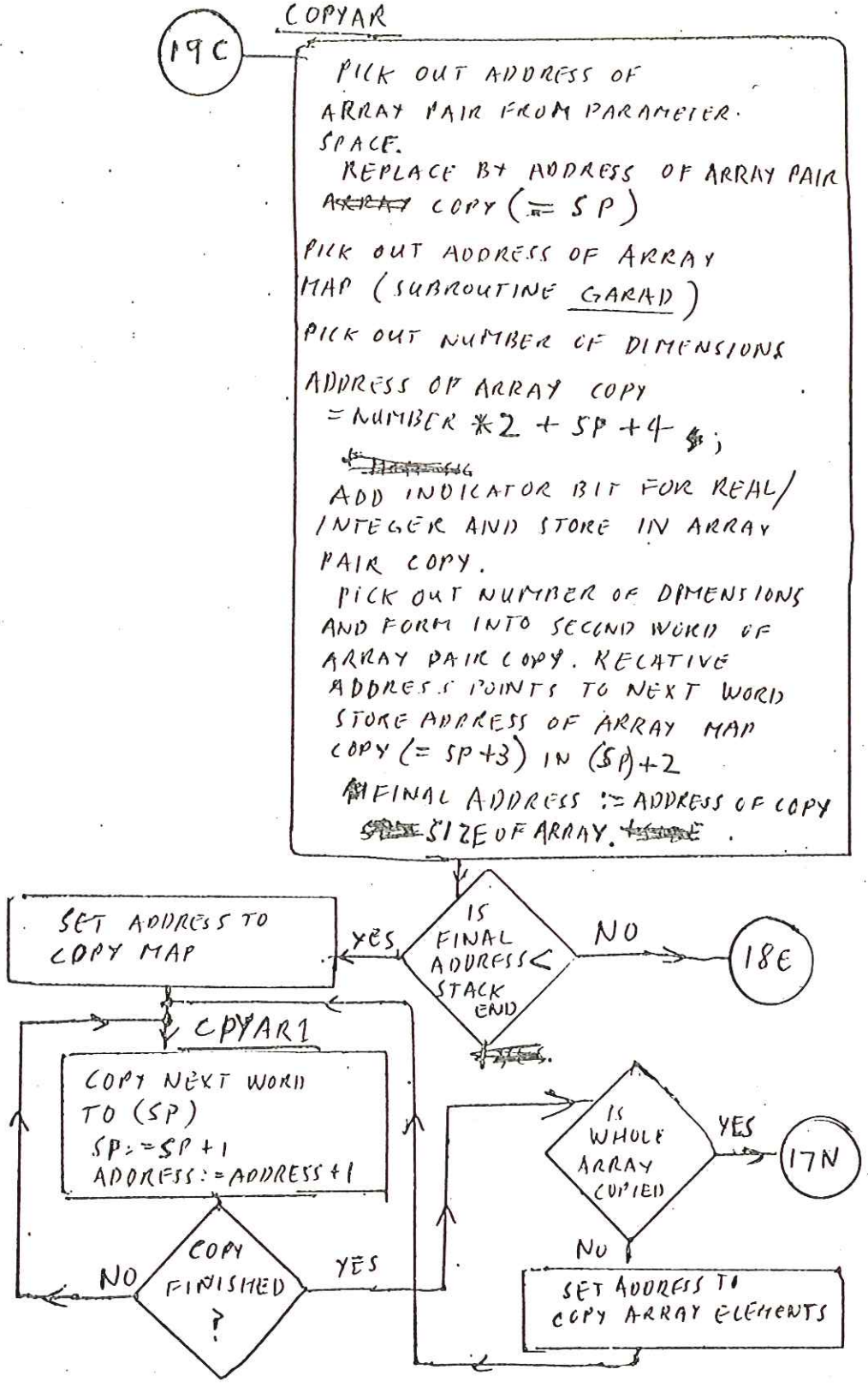

Fun
TIA
14N
ADD QAVNDA ADDRESS
TO ADPART
V:=ADPART
CHECK WORD:=+1
14D
Fun
TRCA
ADD QACODL
ADDRESS TO
ADPART.
ADD (/8 0) TO
ADPART
CHECK WORD:=2
V:=ADPART
14K
Fun
TRA
14R
ADD QAVNDA ADDRESS
TO ADPART
Fun
TICA
14C
Fun
TRAP
14P
ADD MARKER FOR
REAL ADDRESS TO
ADPART. V:=ADPART
TACHK
CHECK WORD:=+2
ADD QACODL
ADDRESS TO
ADPART
ADD (8 0) TO
ADPART
CHECK WORD:=1
V:=ADPART
STORE CHECK WORD
AT SP+1
Fun
TA
14T
(V:=ADPART)
14A
STORE V AT STACK POINTER
(SP)
PRIM
UP
14U
SP:=SP+3
4N
(NXPORD)

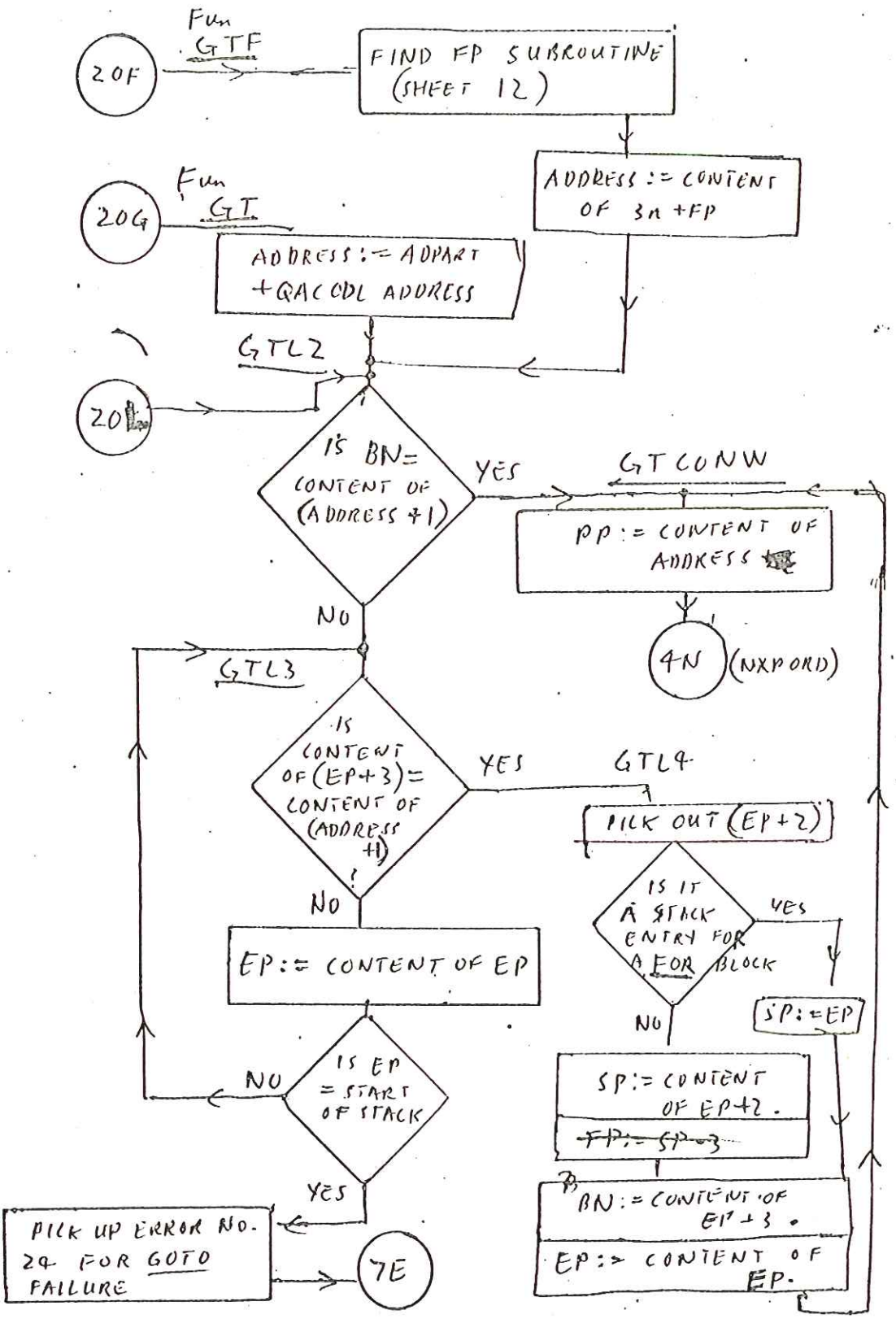

15V
Fun
TIR
ADDRESS := ADPART + QAVNDA ADDRESS
V := CONTENTS OF ADDRESS
14A
15P
Fun
TRR
ADDRESS := ADPART + QAVNDA ADDRESS
15S
STKRLV
STORE CONTENT OF ADDRESS AT SP.
UNPACK M.S. 11 BITS OF CONTENT OF ADDRESS +1
STORE AT SP+1
UNPACK L.S 7 BITS OF CONTENT OF ADDRESS +1
GENERATE SIGN BITS IF BIT 7 IS ONE.
STORE AT SP+2
15K
Fun
TRC
ADDRESS := ADPART + QACODL ADDRESS
14H
15R
Fun
RFUN
SUBROUTINE FINDFP (sheet 12)
V := ADD /0 0 TO 3n+FP
CHECK WORD := /0 2 (REAL, UNPACKED)
15F
IFUN
SUBROUTINE FINDFP (Sheet 12)
V := 3n + FP
CHECK WORD := 1
14K

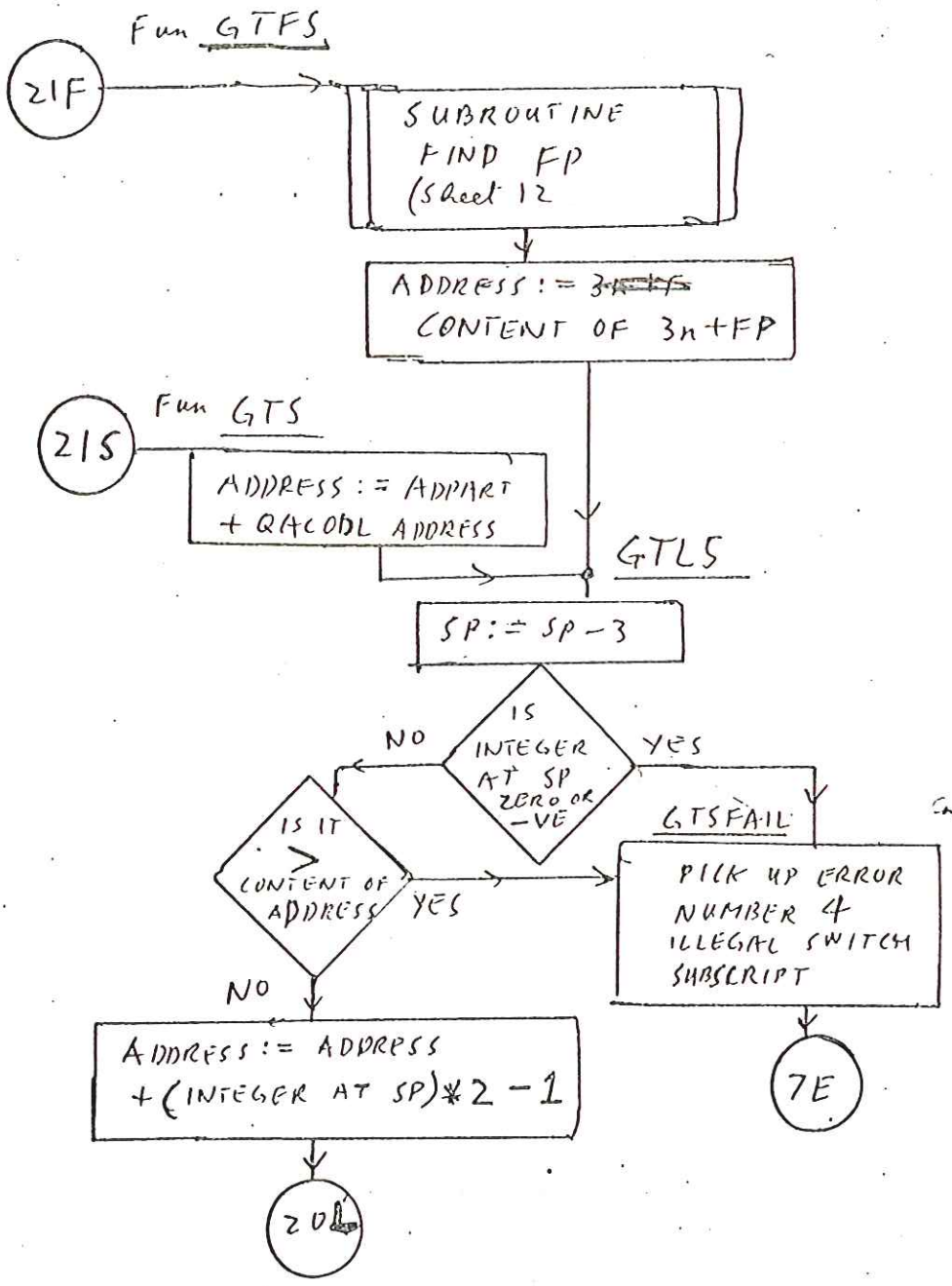

16C
Fun TIC
ADDRESS := ADPART + QACODL ADDRESS
V := CONTENT OF ADDRESS.
14A
16F
Fun IPJ
SP := SP -3
IS VALUE AT (SP) TRUE ?
YES
(NXPORD)
4N
NO
16J
Fun UJ
PP := ADPART
4N
(NXPORD)

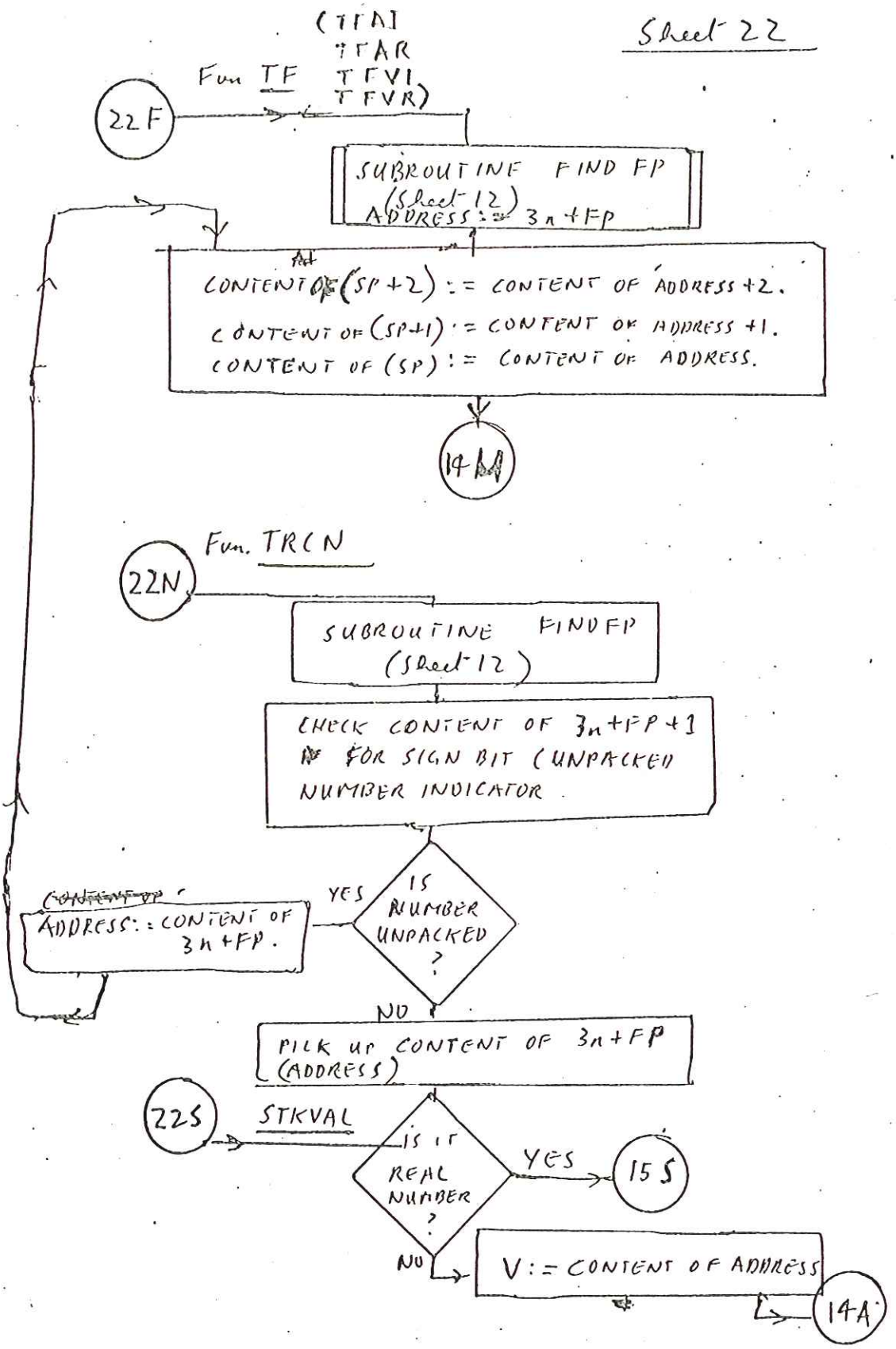

17P
Fun PE
PICK OUT BN' AND
FROM ADPART. NEXTAD :=
CONTENT OF (SP) := EP - 3m
EP := EP - 3m - 3;
CONTENT OF (SP+1) := BN (ORIGINAL);
BN := BN';
SP := SP + 2;
17R
PCHK4
HAVE ALL PROCEDURE PARAMETERS BEEN CHECKED ?
YES
18M
NO
PCHK1
PICK UP NEXT CHECK WORD (CONTENT OF (PP));
PP := PP + 1;
IS PARAM. CALLED BY VALUE ?
YES
PCHK2
IS IT AN ARRAY (INTEGER OR REAL)
YES
19C
NO
17N
NO
PICK UP CODE CHECK FROM (NEXTAD +1) IGNORING M.S. BIT
17M
7E
PICK UP ERROR NO. 1 (PARAMETER MISMATCH)
NO
DOES CHECK WORD CODE CORRESPOND ?
YES
18A
PCHK5
NEXTAD := NEXTAD +3

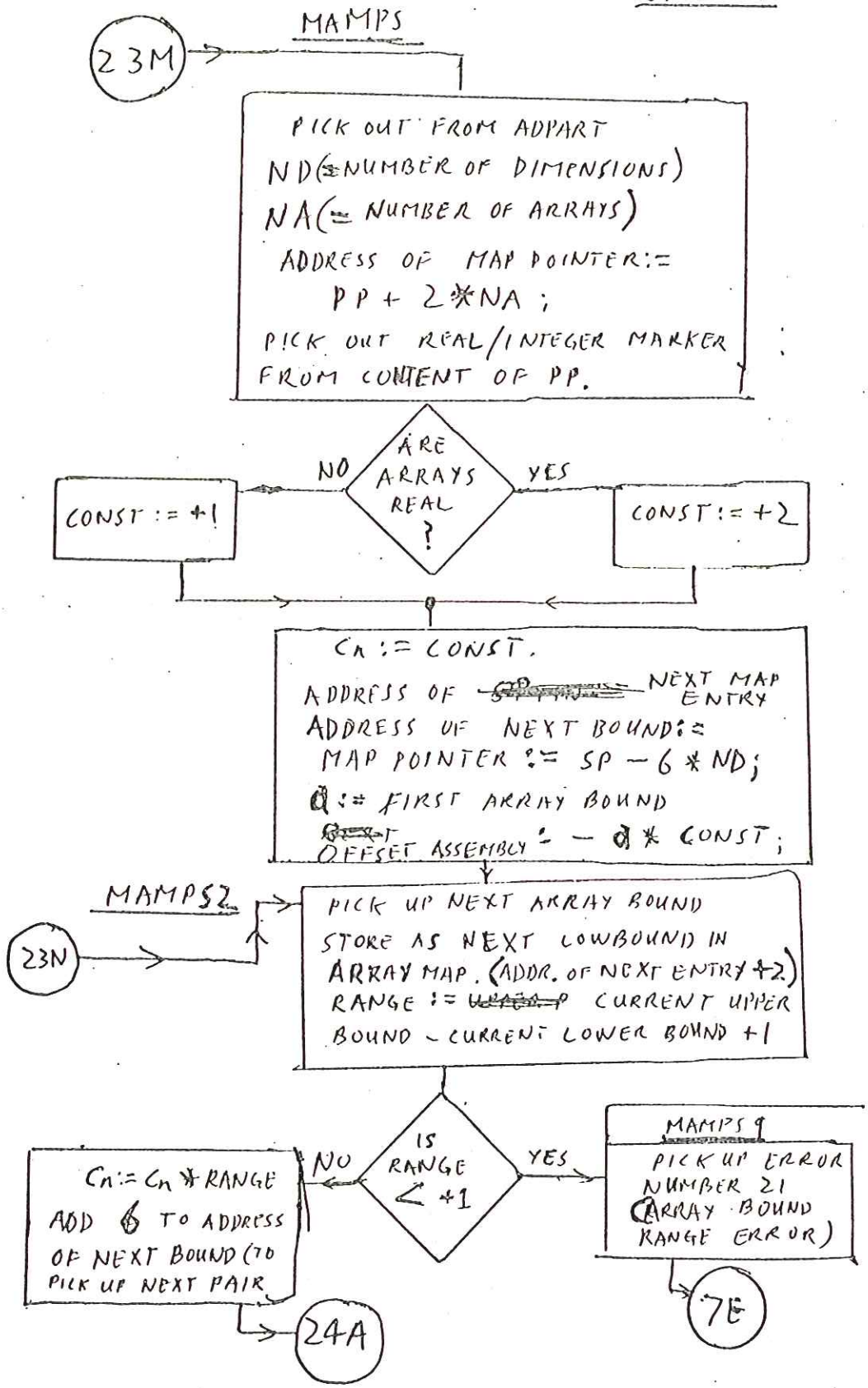

PCHK3
18A
SUBTRACT PROGRAM BASE ADDRESS FROM ADDRESS PART OF CHECK WORD. (RESULT = DIM)
WAS ADDRESS PART = 8191
YES
17N
(IGNORE IF PARAMETER NOT USED IN PROCEDURE)
NO
PCHK8
IS PARAMETER AN ARRAY
YES
PICK OUT NUMBER OF DIMENSIONS
NO
IS IT A PROCEDURE
PCHK8
YES
PICK OUT NUMBER OF PARAMETERS FROM BLOCK HEAD
NO
IS NUMBER = DIM ?
YES
17N
NO
17M
18M
MAMPS4
IS SP < WARN ADDRESS ?
YES
4N
(NXPORD)
NO
SET WARN TRUE
18E
MAMPS5
IS SP < STACK END
NO
YES
4N
ERROR NUMBER 2 STACK OVERFLOW
7E

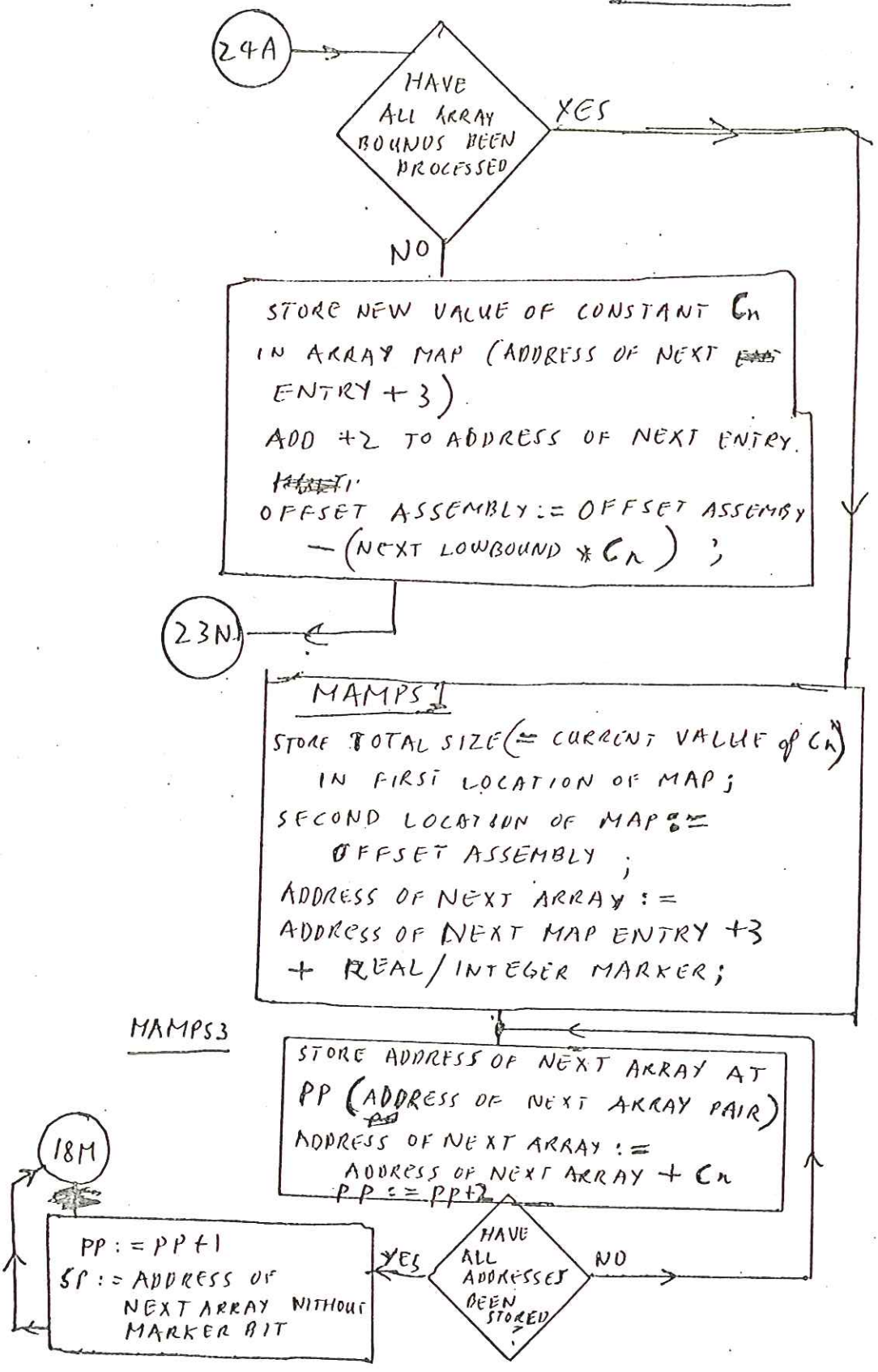

19C
COPYAR
PICK OUT ADDRESS OF ARRAY PAIR FROM PARAMETER SPACE.
REPLACE BY ADDRESS OF ARRAY PAIR COPY (= SP)
PICK OUT ADDRESS OF ARRAY MAP (SUBROUTINE GARAD)
PICK OUT NUMBER OF DIMENSIONS
ADDRESS OF ARRAY COPY = NUMBER *2 + SP + 4;
ADD INDICATOR BIT FOR REAL/INTEGER AND STORE IN ARRAY PAIR COPY.
PICK OUT NUMBER OF DIMENSIONS AND FORM INTO SECOND WORD OF ARRAY PAIR COPY. RELATIVE ADDRESS POINTS TO NEXT WORD
STORE ADDRESS OF ARRAY MAP COPY (= SP+3) IN (SP)+2
FINAL ADDRESS := ADDRESS OF COPY SIZE OF ARRAY.
IS FINAL ADDRESS < STACK END
YES
NO
18E
SET ADDRESS TO COPY MAP
COPYAR1
COPY NEXT WORD TO (SP)
SP := SP + 1
ADDRESS := ADDRESS + 1
COPY FINISHED ?
NO
YES
IS WHOLE ARRAY COPIED
YES
17N
NO
SET ADDRESS TO COPY ARRAY ELEMENTS


20F
Fun GTF
FIND FP SUBROUTINE (SHEET 12)
ADDRESS := CONTENT OF 3n + FP
20G
Fun GT
ADDRESS := ADPART + QACODL ADDRESS
GTL2
20B
IS BN = CONTENT OF (ADDRESS + 1)
YES
GT CONW
pp := CONTENT OF ADDRESS
4N (NXPORD)
NO
GTL3
IS CONTENT OF (EP+3) = CONTENT OF (ADDRESS + 1)
YES
GTL4
PICK OUT (EP+2)
IS IT A STACK ENTRY FOR A FOR BLOCK
YES
SP := EP
NO
SP := CONTENT OF EP+2.
FP := SP - 3
BN := CONTENT OF EP + 3.
EP := CONTENT OF EP.
NO
EP := CONTENT OF EP
IS EP = START OF STACK
NO
YES
PICK UP ERROR NO. 24 FOR GOTO FAILURE
7E

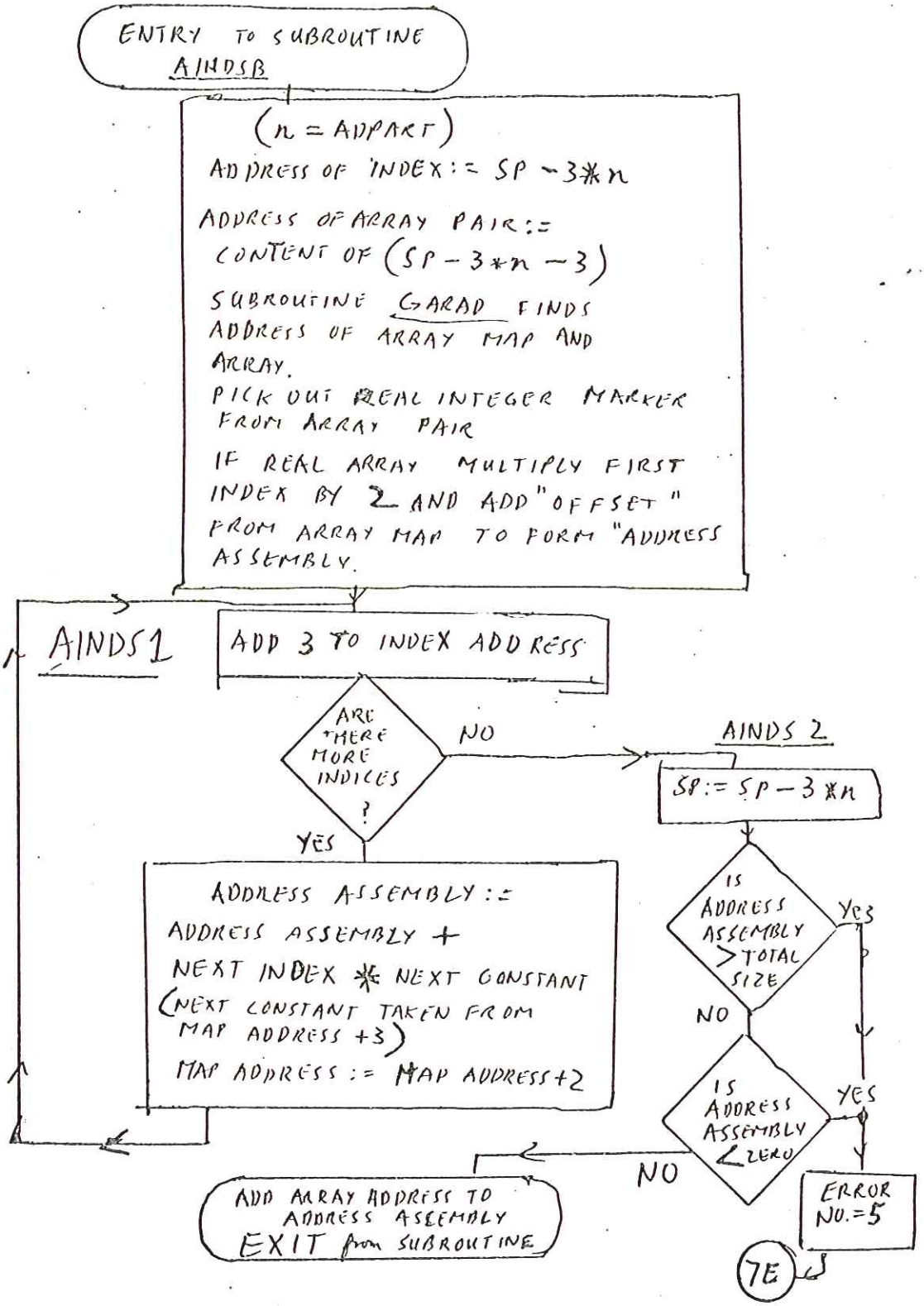

Fun GTFS
21F
SUBROUTINE FIND FP (Sheet 12
ADDRESS := CONTENT OF 3n+FP
Fun GTS
21S
ADDRESS := ADPART + QACODL ADDRESS
GTL5
SP := SP-3
IS INTEGER AT SP ZERO OR -VE
NO
YES
IS IT > CONTENT OF ADDRESS
YES
NO
GTSFAIL
PICK UP ERROR NUMBER 4 ILLEGAL SWITCH SUBSCRIPT
7E
ADDRESS := ADDRESS + (INTEGER AT SP)*2 - 1
20L

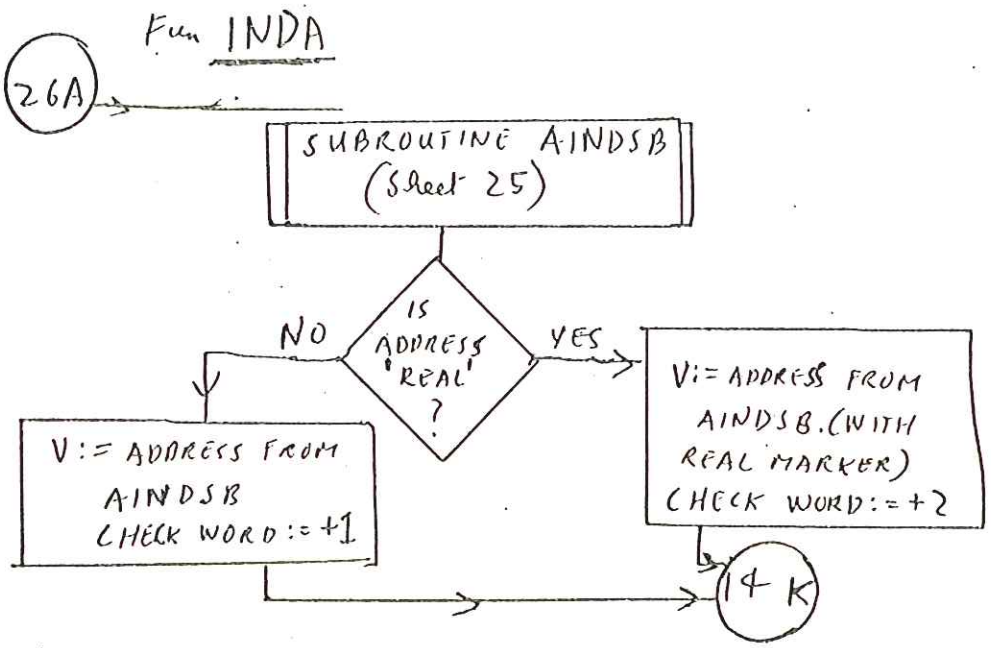

(TFA
TFAR
TFVI
TFVR)
Fun TF
22F
SUBROUTINE FIND FP
(Sheet 12)
ADDRESS := 3n+FP
CONTENT OF (SP+2) := CONTENT OF ADDRESS+2.
CONTENT OF (SP+1) := CONTENT OF ADDRESS+1.
CONTENT OF (SP) := CONTENT OF ADDRESS.
14M
Fun. TRCN
22N
SUBROUTINE FINDFP
(Sheet 12)
CHECK CONTENT OF 3n+FP+1
FOR SIGN BIT (UNPACKED
NUMBER INDICATOR
IS NUMBER UNPACKED ?
YES
CONTENT OF
ADDRESS := CONTENT OF
3n+FP.
NO
PICK UP CONTENT OF 3n+FP
(ADDRESS)
22S
STKVAL
IS IT REAL NUMBER ?
YES
15S
NO
V := CONTENT OF ADDRESS
14A

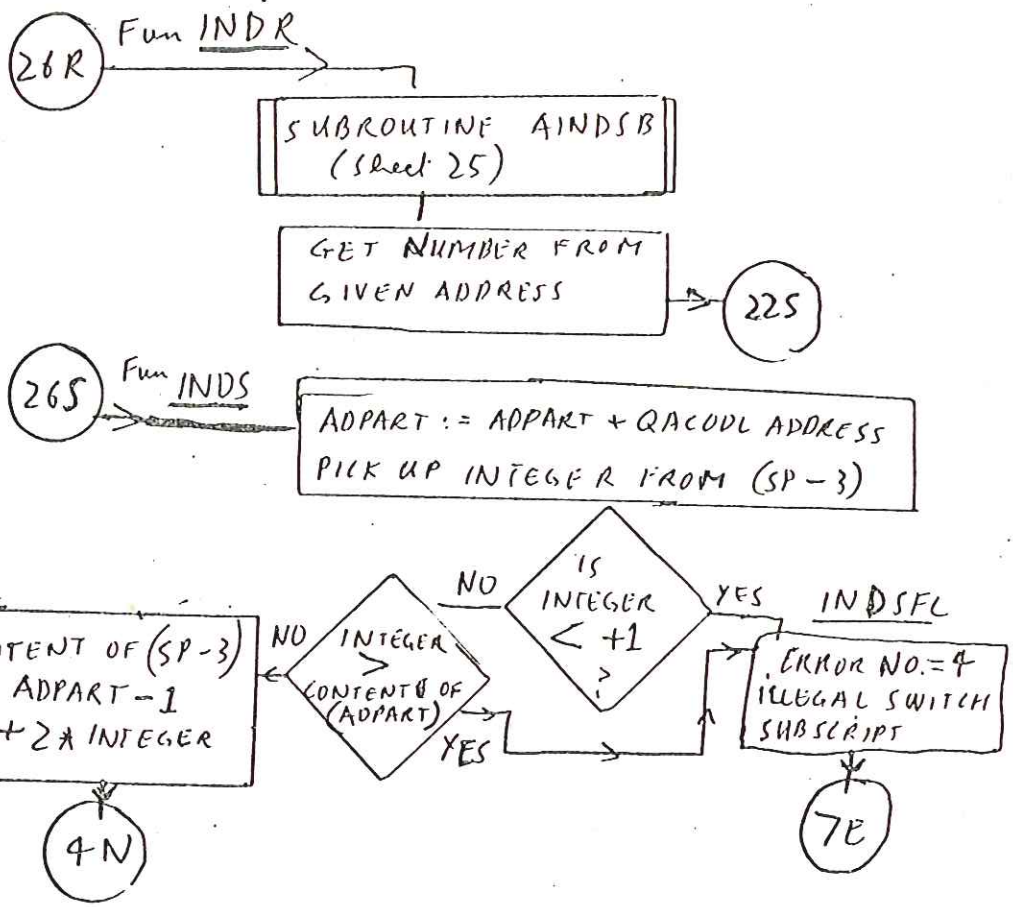

23M
MAMPS
PICK OUT FROM ADPART
ND(=NUMBER OF DIMENSIONS)
NA(= NUMBER OF ARRAYS)
ADDRESS OF MAP POINTER:=
PP + 2*NA ;
PICK OUT REAL/INTEGER MARKER
FROM CONTENT OF PP.
ARE ARRAYS REAL ?
NO
YES
CONST := +1
CONST := +2
Cn := CONST.
ADDRESS OF NEXT MAP ENTRY
ADDRESS OF NEXT BOUND:=
MAP POINTER := SP - 6 * ND;
Q := FIRST ARRAY BOUND
OFFSET ASSEMBLY := - Q * CONST;
MAMPS2
23N
PICK UP NEXT ARRAY BOUND
STORE AS NEXT LOWBOUND IN
ARRAY MAP. (ADDR. OF NEXT ENTRY +2)
RANGE := CURRENT UPPER
BOUND - CURRENT LOWER BOUND +1
IS RANGE < +1
NO
YES
Cn := Cn * RANGE
ADD 6 TO ADDRESS
OF NEXT BOUND (TO
PICK UP NEXT PAIR
24A
MAMPS 9
PICK UP ERROR
NUMBER 21
(ARRAY BOUND
RANGE ERROR)
7E

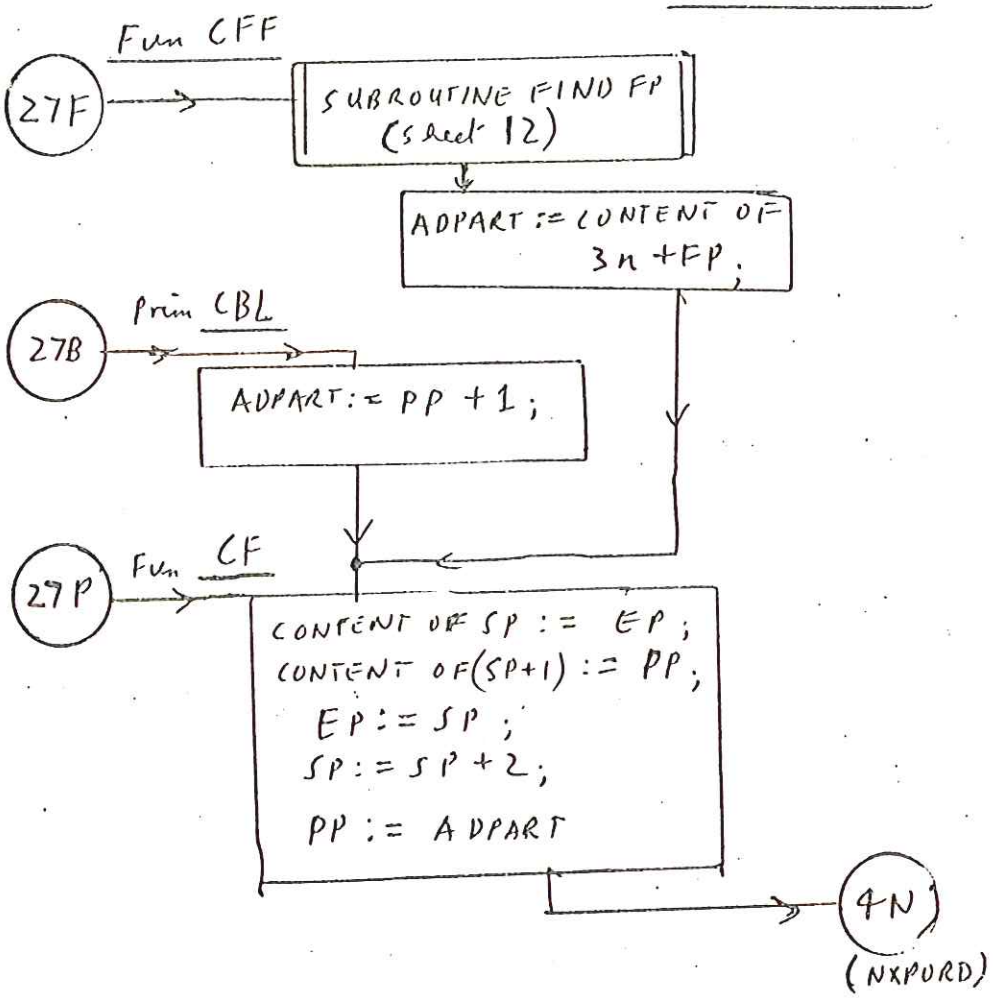

24A
HAVE ALL ARRAY BOUNDS BEEN PROCESSED
YES
NO
STORE NEW VALUE OF CONSTANT Cn IN ARRAY MAP (ADDRESS OF NEXT ENTRY + 3)
ADD +2 TO ADDRESS OF NEXT ENTRY.
OFFSET ASSEMBLY := OFFSET ASSEMBLY − (NEXT LOWBOUND * Cn);
23N
MAMPS1
STORE TOTAL SIZE (= CURRENT VALUE of Cn) IN FIRST LOCATION OF MAP;
SECOND LOCATION OF MAP := OFFSET ASSEMBLY;
ADDRESS OF NEXT ARRAY := ADDRESS OF NEXT MAP ENTRY +3 + REAL/INTEGER MARKER;
MAMPS3
STORE ADDRESS OF NEXT ARRAY AT PP (ADDRESS OF NEXT ARRAY PAIR)
ADDRESS OF NEXT ARRAY := ADDRESS OF NEXT ARRAY + Cn
PP := PP+2
HAVE ALL ADDRESSES BEEN STORED?
YES
NO
PP := PP+1
SP := ADDRESS OF NEXT ARRAY WITHOUT MARKER BIT
18M

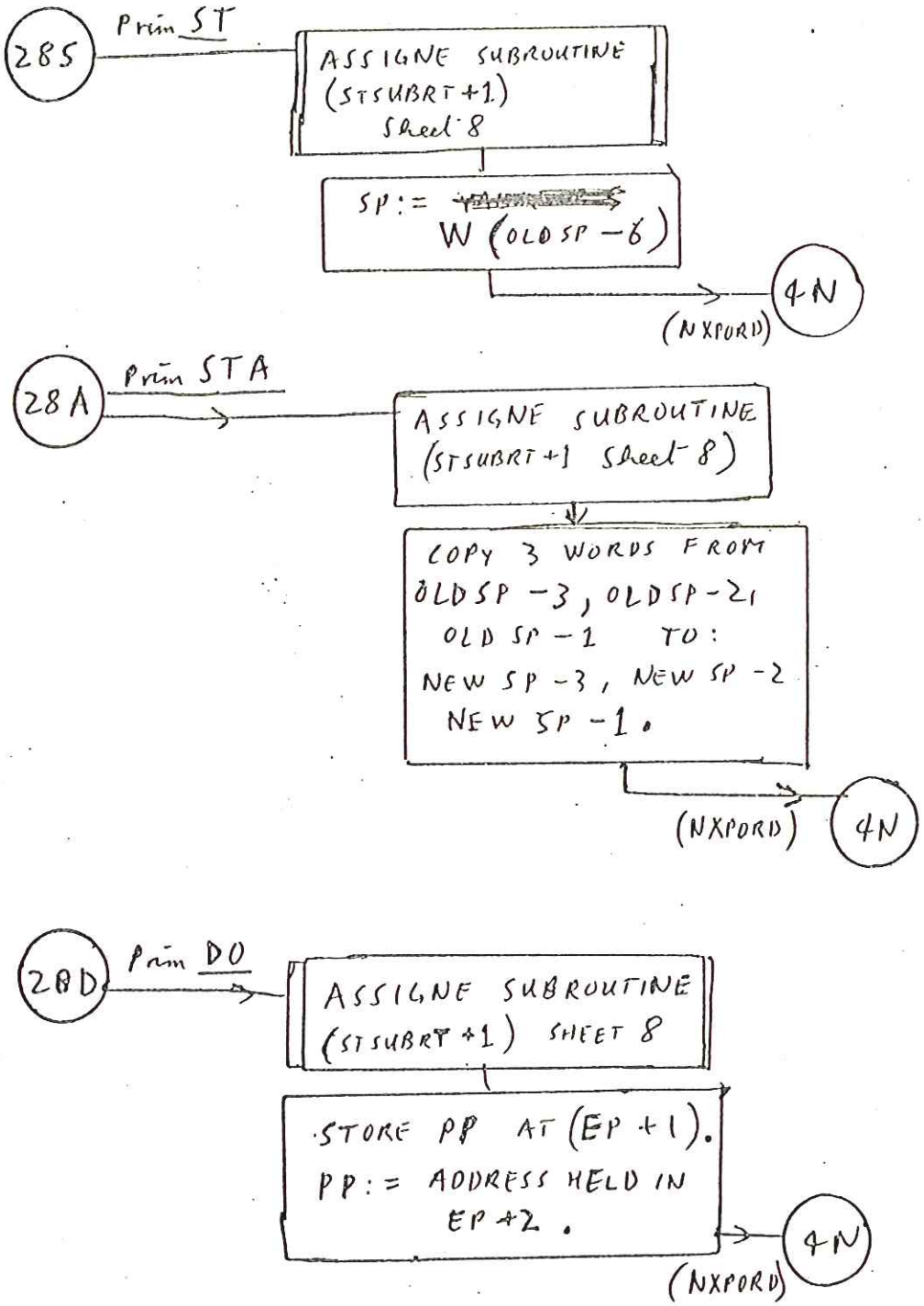
# SUBROUTINE : AINDSB

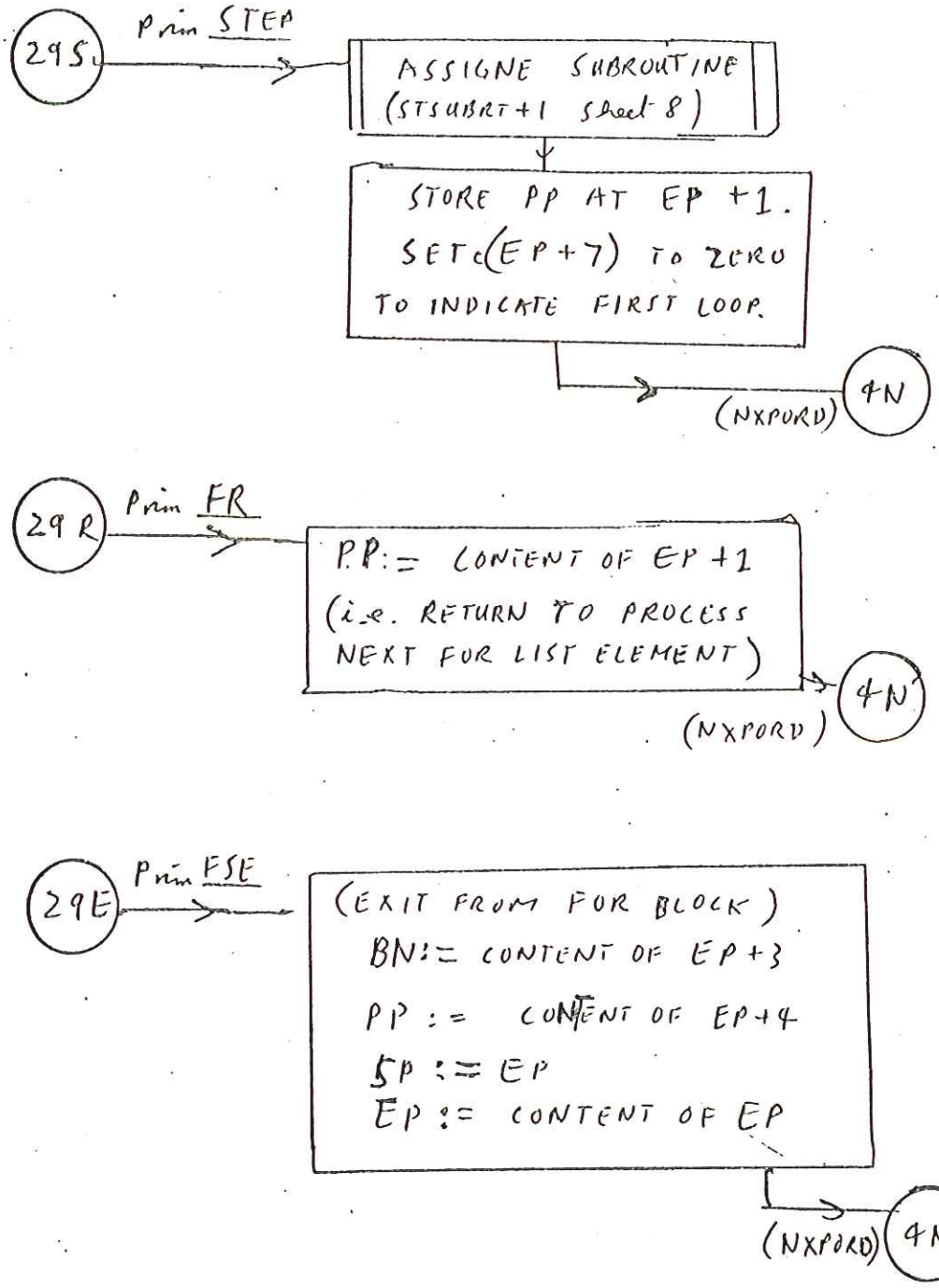


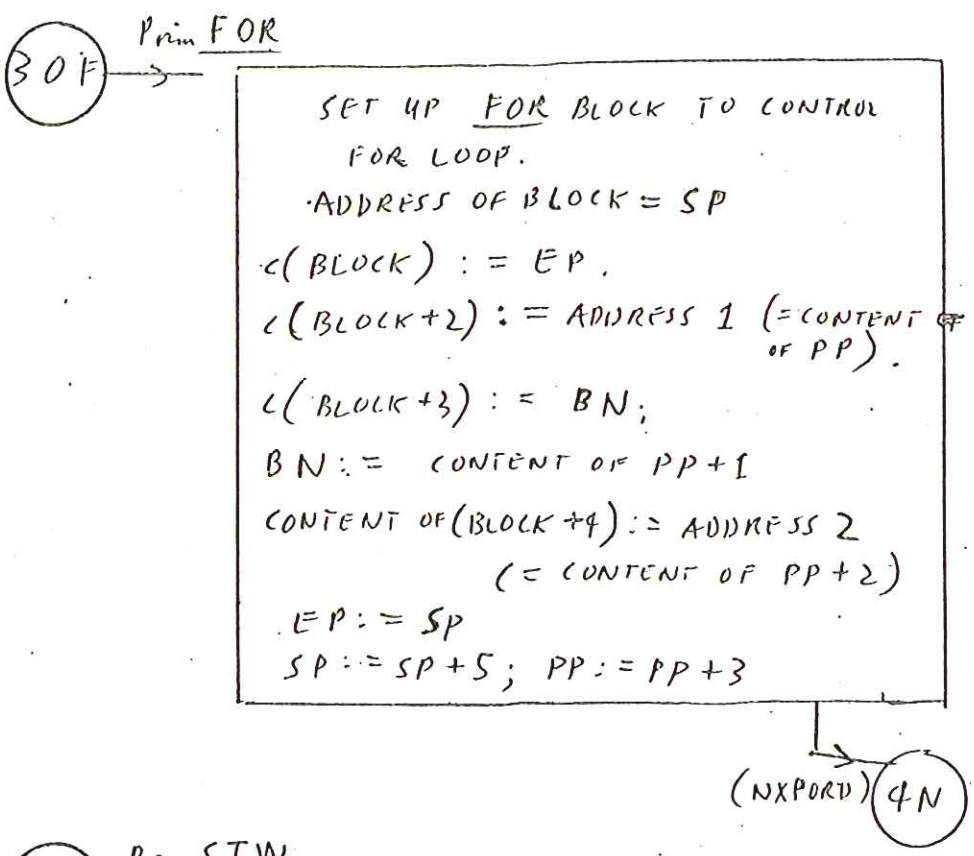

Fun INDA
26A
SUBROUTINE AINDSB (Sheet 25)
IS ADDRESS 'REAL' ?
NO
YES
V:= ADDRESS FROM AINDSB. (WITH REAL MARKER) CHECK WORD:= +2
V:= ADDRESS FROM AINDSB CHECK WORD:= +1
14K

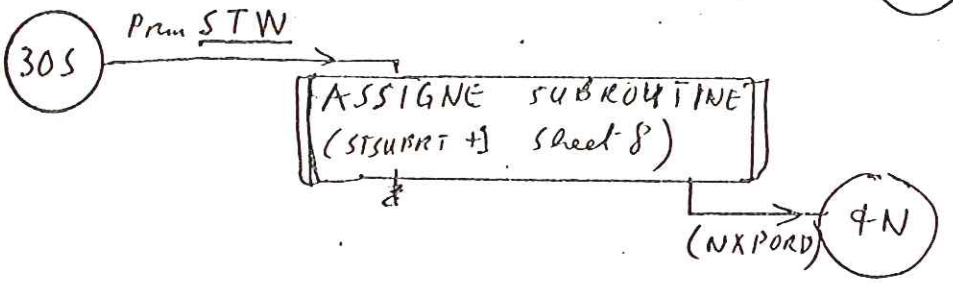

26R
Fun INDR
SUBROUTINE AINDSB (Sheet 25)
GET NUMBER FROM GIVEN ADDRESS
22S
26S
Fun INDS
ADPART := ADPART + QACODL ADDRESS
PICK UP INTEGER FROM (SP-3)
IS INTEGER < +1 ?
NO
YES
INDSFL
ERROR NO.=4 ILLEGAL SWITCH SUBSCRIPT
7E
INTEGER > CONTENTS OF (ADPART)
NO
YES
CONTENT OF (SP-3) := ADPART-1 +2* INTEGER
4N

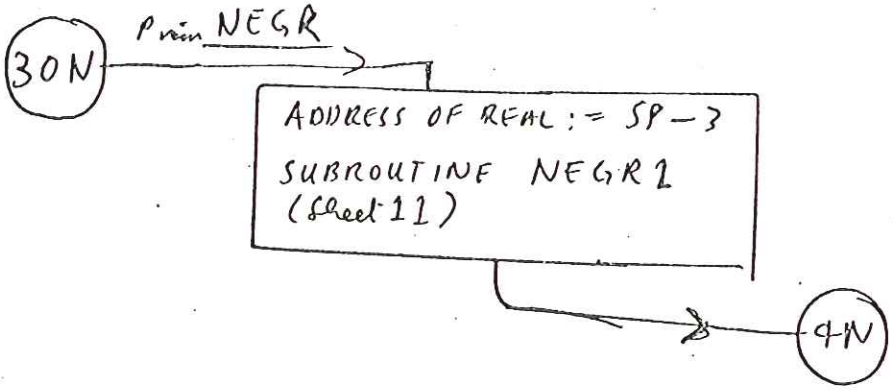

Fun CFF
27F
SUBROUTINE FIND FP
(sheet 12)
ADPART := CONTENT OF
3n + FP;
Prim CBL
27B
ADPART := PP + 1;
27P
Fun CF
CONTENT OF SP := EP;
CONTENT OF (SP+1) := PP;
EP := SP;
SP := SP + 2;
PP := ADPART
4N
(NXPORD)

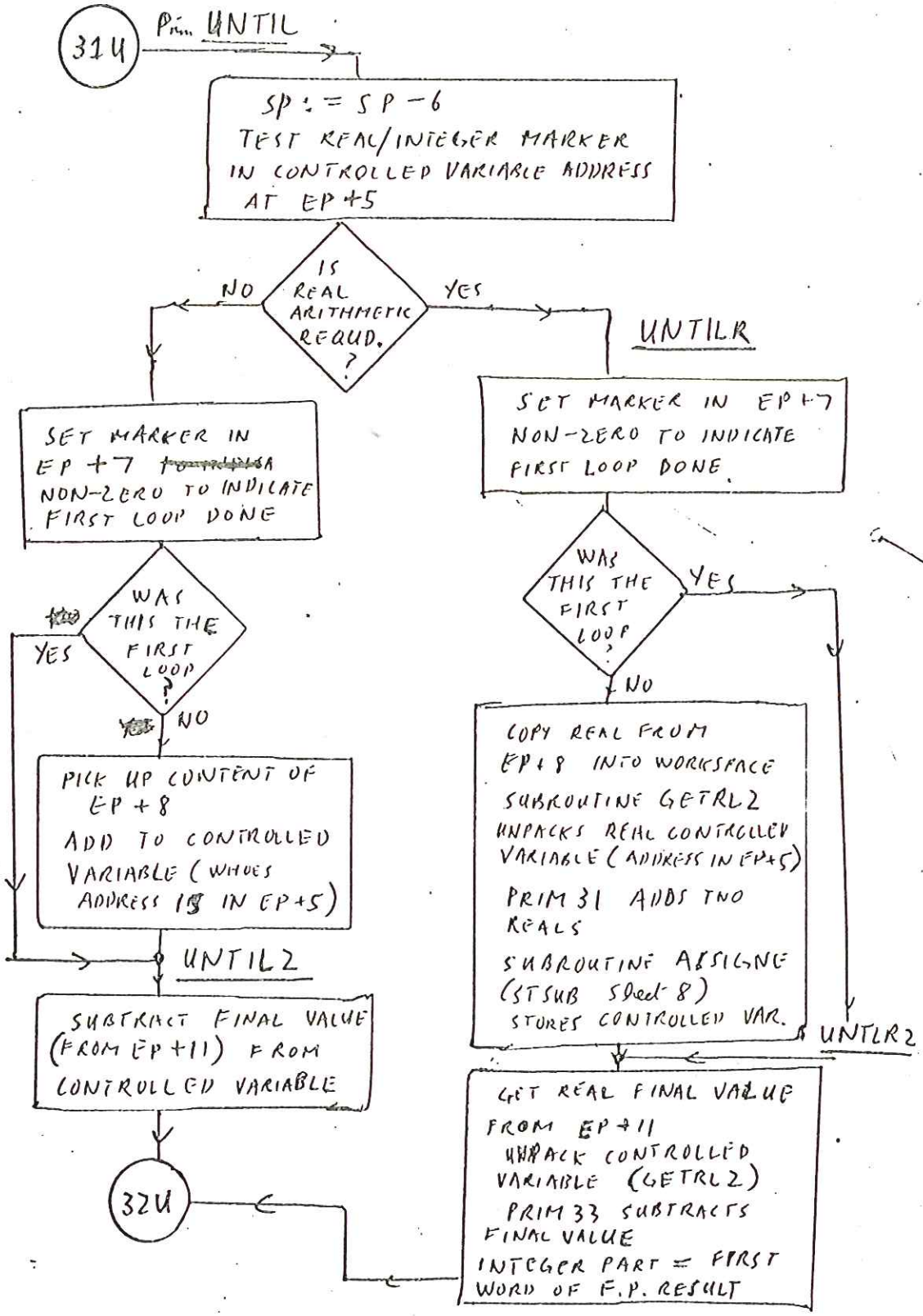

28S
Prim ST
ASSIGNE SUBROUTINE
(STSUBRT+1)
Sheet 8
SP := W (OLDSP - 6)
4N
(NXPORD)
28A
Prim STA
ASSIGNE SUBROUTINE
(STSUBRT+1 Sheet 8)
COPY 3 WORDS FROM
OLDSP - 3, OLDSP - 2,
OLD SP - 1 TO:
NEW SP - 3, NEW SP - 2
NEW SP - 1.
(NXPORD)
4N
28D
Prim DO
ASSIGNE SUBROUTINE
(STSUBRT+1) SHEET 8
STORE PP AT (EP+1).
PP := ADDRESS HELD IN
EP+2.
4N
(NXPORD)

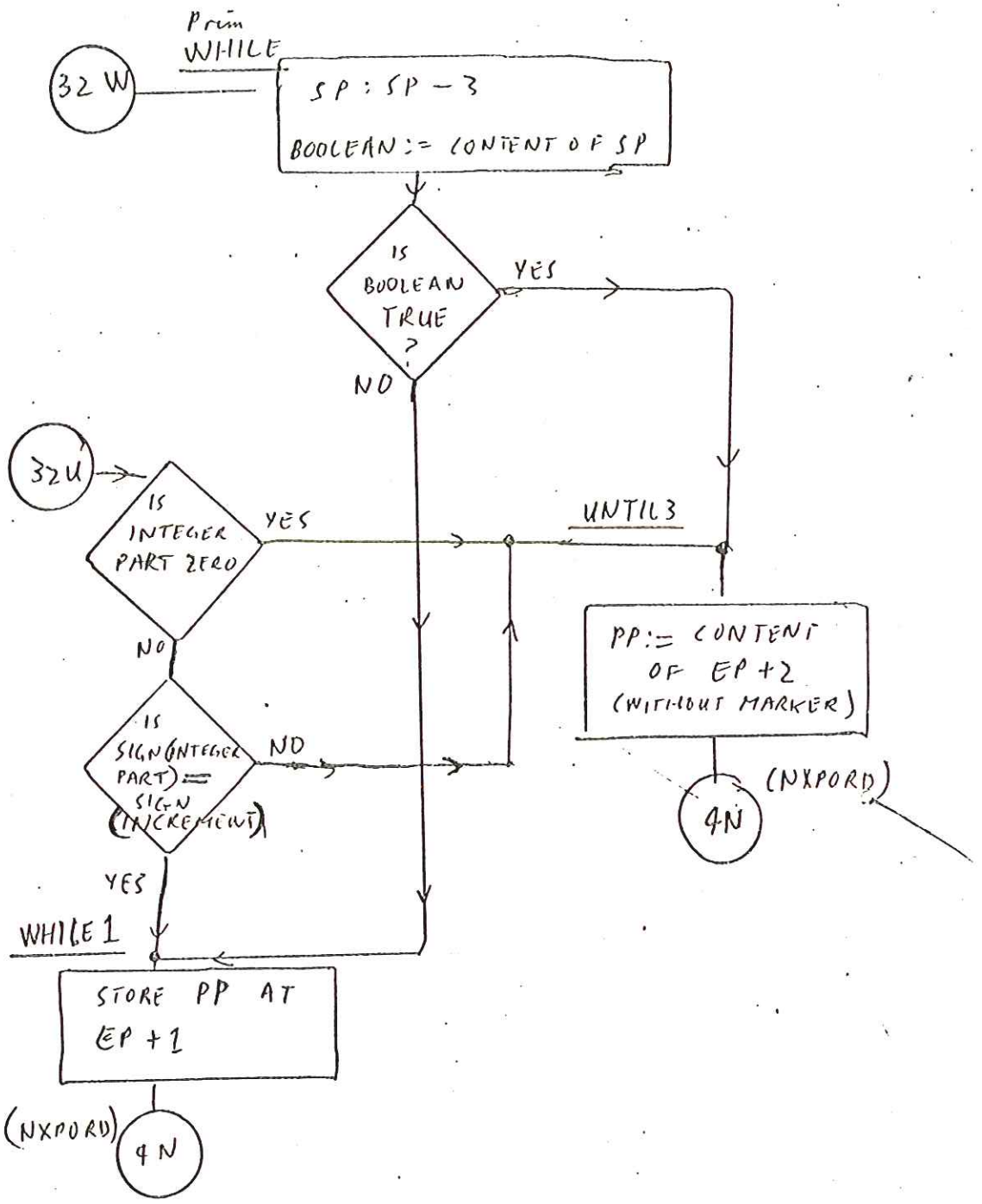

29S
Prim STEP
ASSIGNE SUBROUTINE
(STSUBRT+1 Sheet 8)
STORE PP AT EP +1.
SET (EP+7) TO ZERO
TO INDICATE FIRST LOOP.
(NXPORD)
4N
29R
Prim FR
PP:= CONTENT OF EP+1
(i.e. RETURN TO PROCESS
NEXT FOR LIST ELEMENT)
(NXPORD)
4N
29E
Prim FSE
(EXIT FROM FOR BLOCK)
BN:= CONTENT OF EP+3
PP := CONTENT OF EP+4
SP := EP
EP := CONTENT OF EP
(NXPORD)
4N

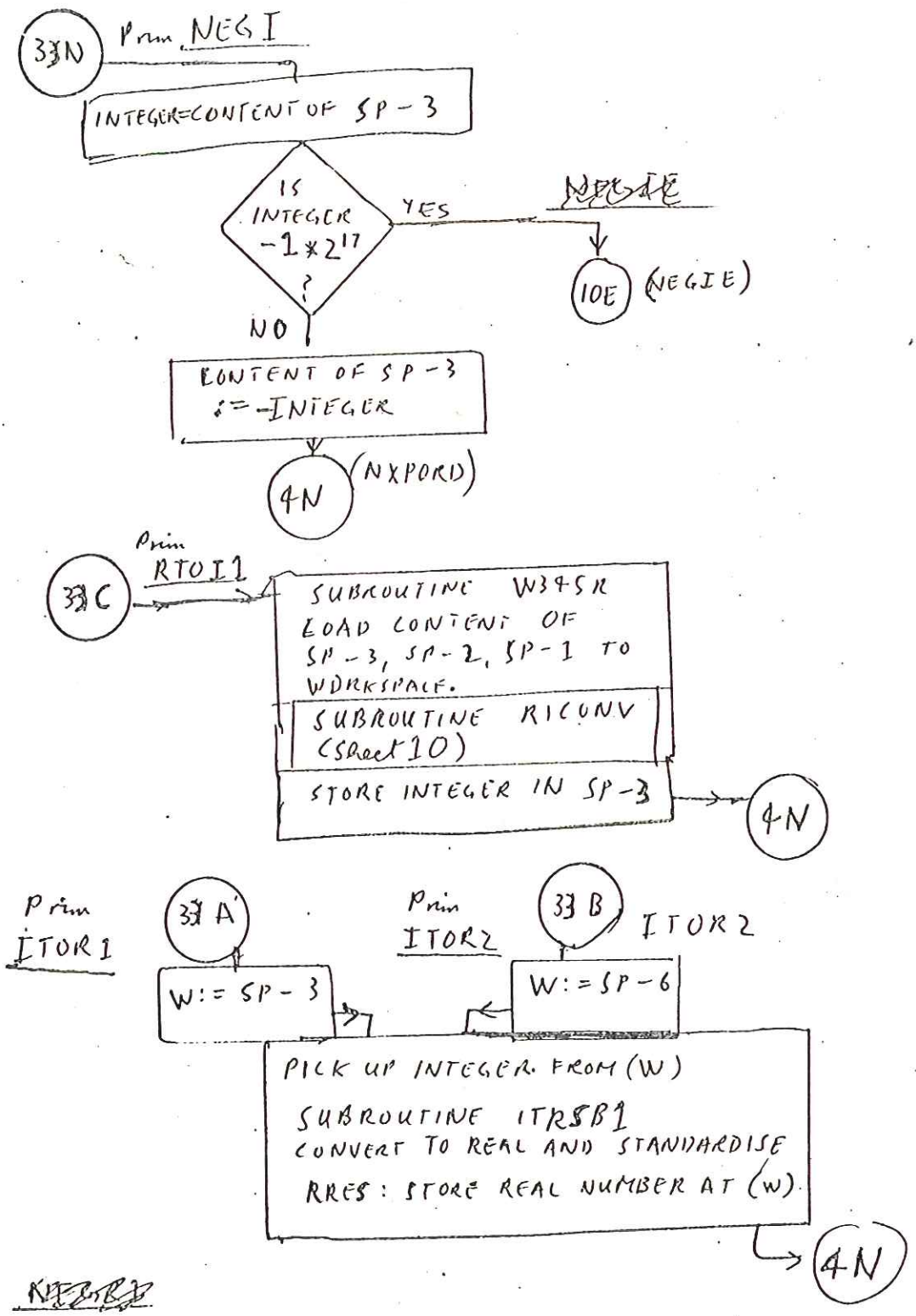

30F
Prim FOR
SET UP FOR BLOCK TO CONTROL FOR LOOP.
ADDRESS OF BLOCK = SP
c(BLOCK) := EP.
c(BLOCK+2) := ADDRESS 1 (= CONTENT OF PP).
c(BLOCK+3) := BN;
BN := CONTENT OF PP+1
CONTENT OF (BLOCK+4) := ADDRESS 2 (= CONTENT OF PP+2)
EP := SP
SP := SP+5; PP := PP+3
(NXPORD)
4N

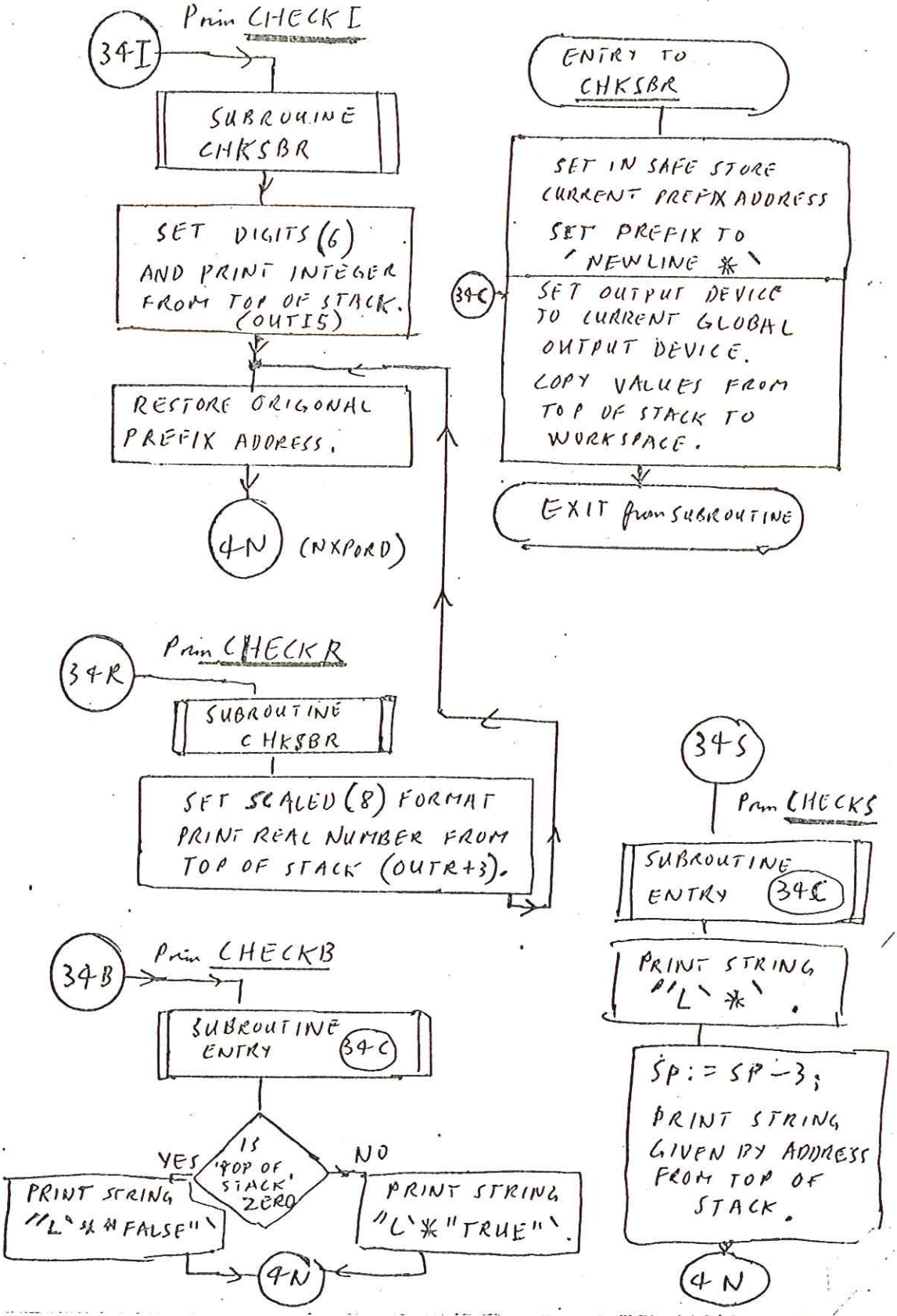

30S
Prim STW
ASSIGNE SUBROUTINE (STSUBRT +1 Sheet 8)
(NXPORD)
4N

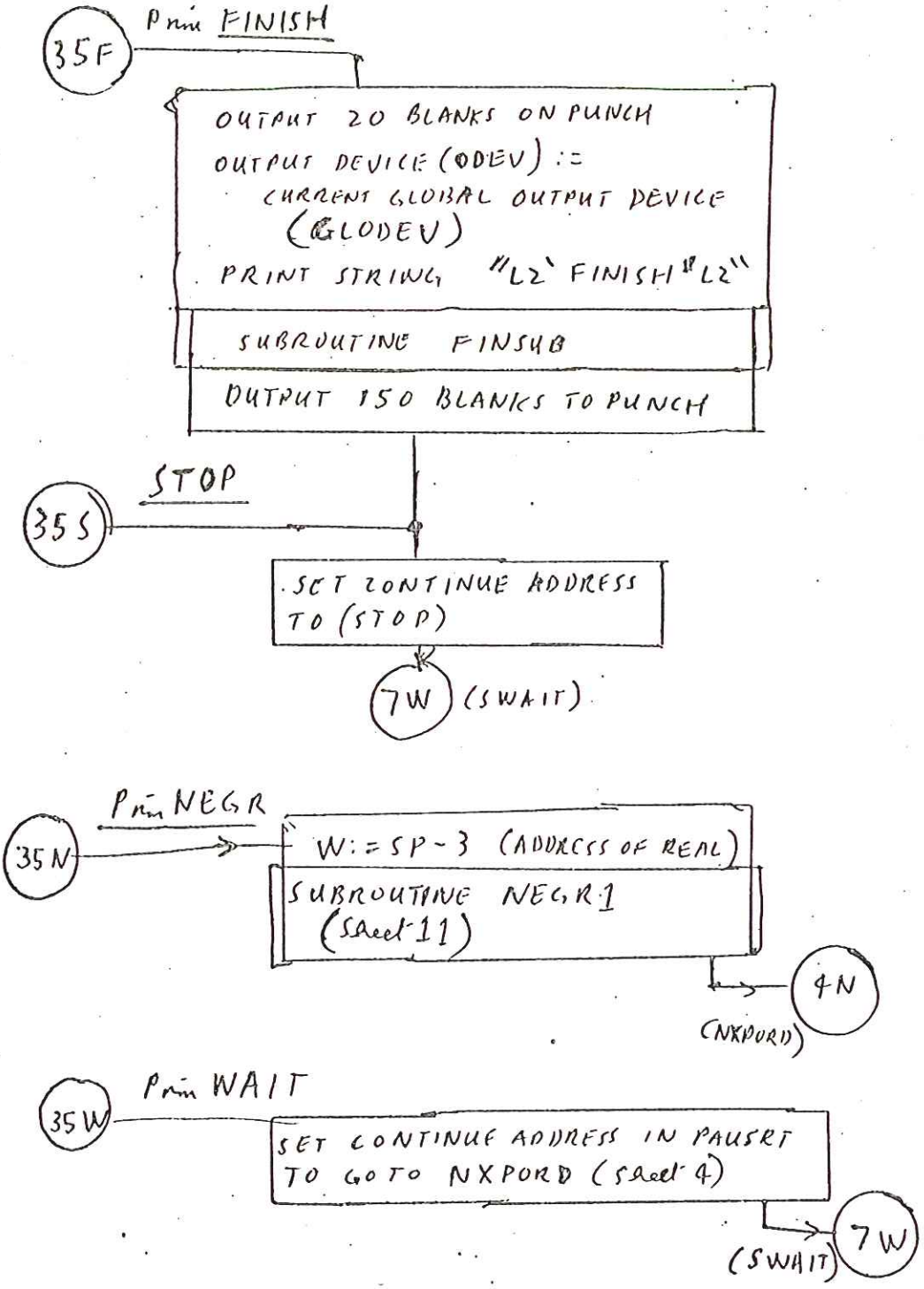

30N
Prim NEGR
ADDRESS OF REAL := SP-3
SUBROUTINE NEGR1 (Sheet 11)
4N

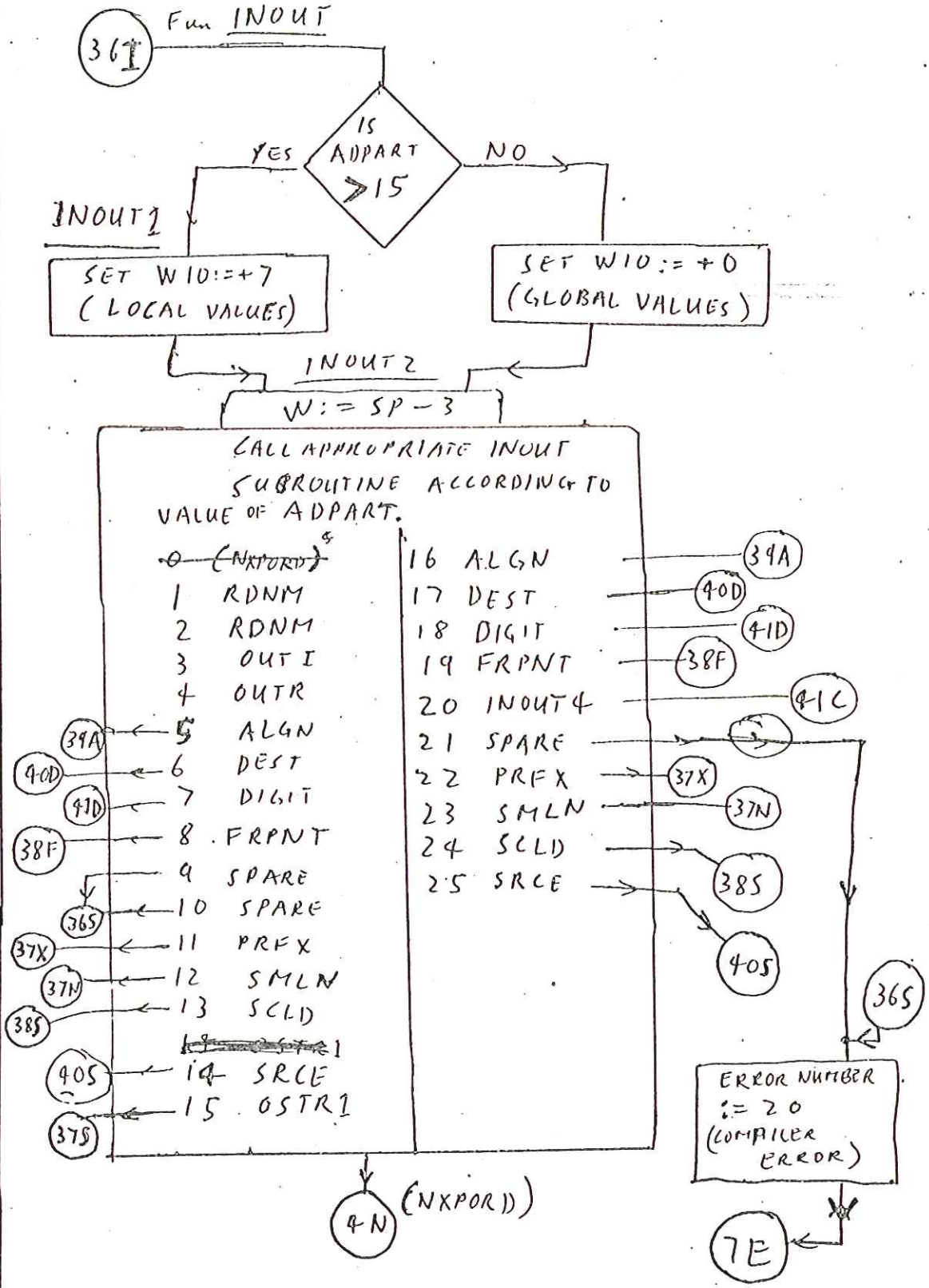

314
Prim UNTIL
SP := SP - 6
TEST REAL/INTEGER MARKER IN CONTROLLED VARIABLE ADDRESS AT EP+5
IS REAL ARITHMETIC REQUD. ?
NO
YES
UNTILR
SET MARKER IN EP+7 NON-ZERO TO INDICATE FIRST LOOP DONE
SET MARKER IN EP+7 NON-ZERO TO INDICATE FIRST LOOP DONE
WAS THIS THE FIRST LOOP ?
YES
NO
WAS THIS THE FIRST LOOP ?
YES
NO
PICK UP CONTENT OF EP+8
ADD TO CONTROLLED VARIABLE (WHOES ADDRESS IS IN EP+5)
COPY REAL FROM EP+8 INTO WORKSPACE
SUBROUTINE GETRL2 UNPACKS REAL CONTROLLED VARIABLE (ADDRESS IN EP+5)
PRIM 31 ADDS TWO REALS
SUBROUTINE ASSIGNE (STSUB Sheet 8) STORES CONTROLLED VAR.
UNTIL2
UNTILR2
SUBTRACT FINAL VALUE (FROM EP+11) FROM CONTROLLED VARIABLE
GET REAL FINAL VALUE FROM EP+11
UNPACK CONTROLLED VARIABLE (GETRL2)
PRIM 33 SUBTRACTS FINAL VALUE
INTEGER PART = FIRST WORD OF F.P. RESULT
32U

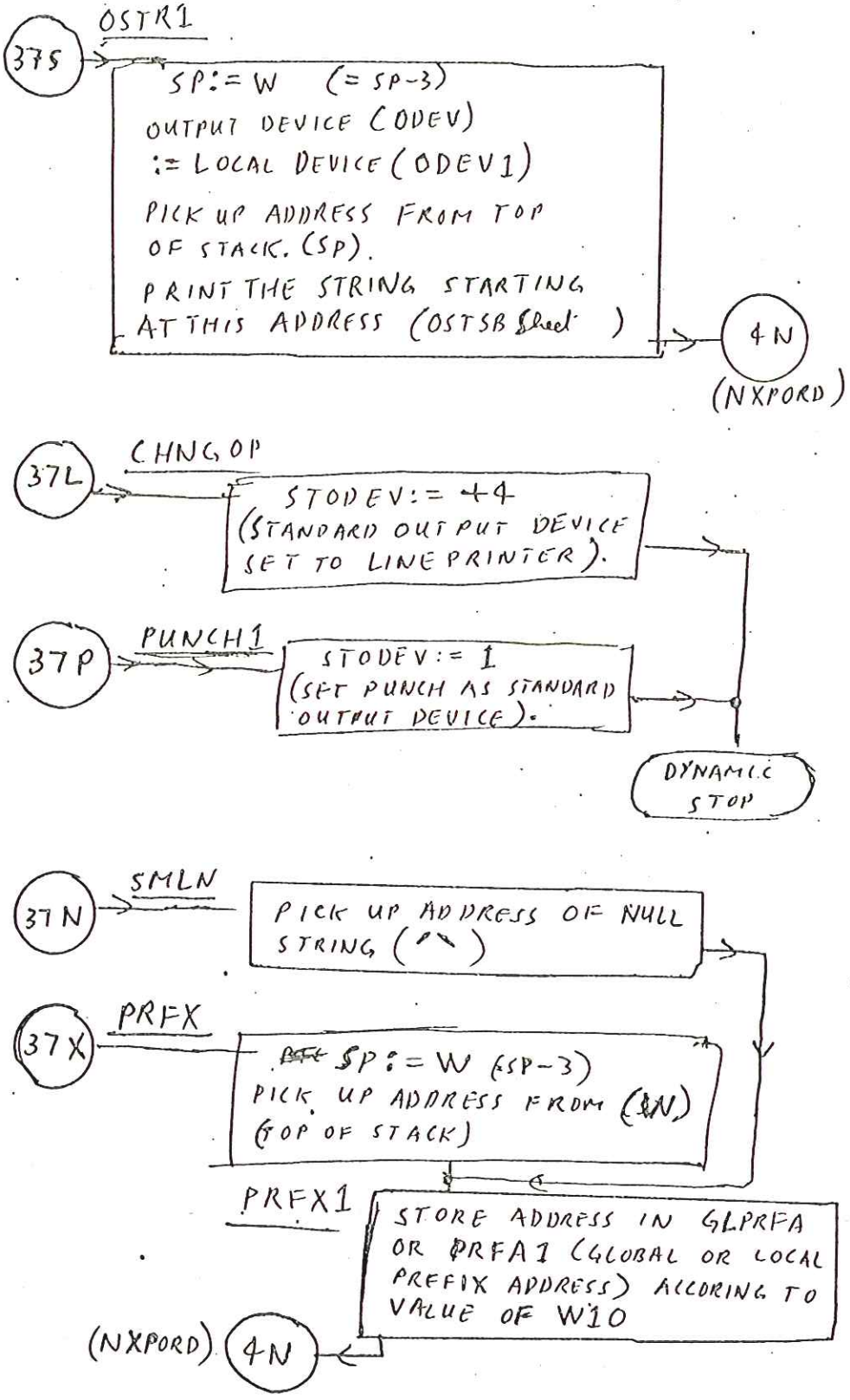

Prim WHILE
32 W
SP : SP − 3
BOOLEAN := CONTENT OF SP
IS BOOLEAN TRUE ?
YES
NO
32 U
IS INTEGER PART ZERO
YES
UNTIL3
NO
IS SIGN(INTEGER PART) = SIGN (INCREMENT)
NO
YES
WHILE 1
PP := CONTENT OF EP + 2 (WITHOUT MARKER)
(NXPORD)
4N
STORE PP AT EP + 1
(NXPORD)
4 N

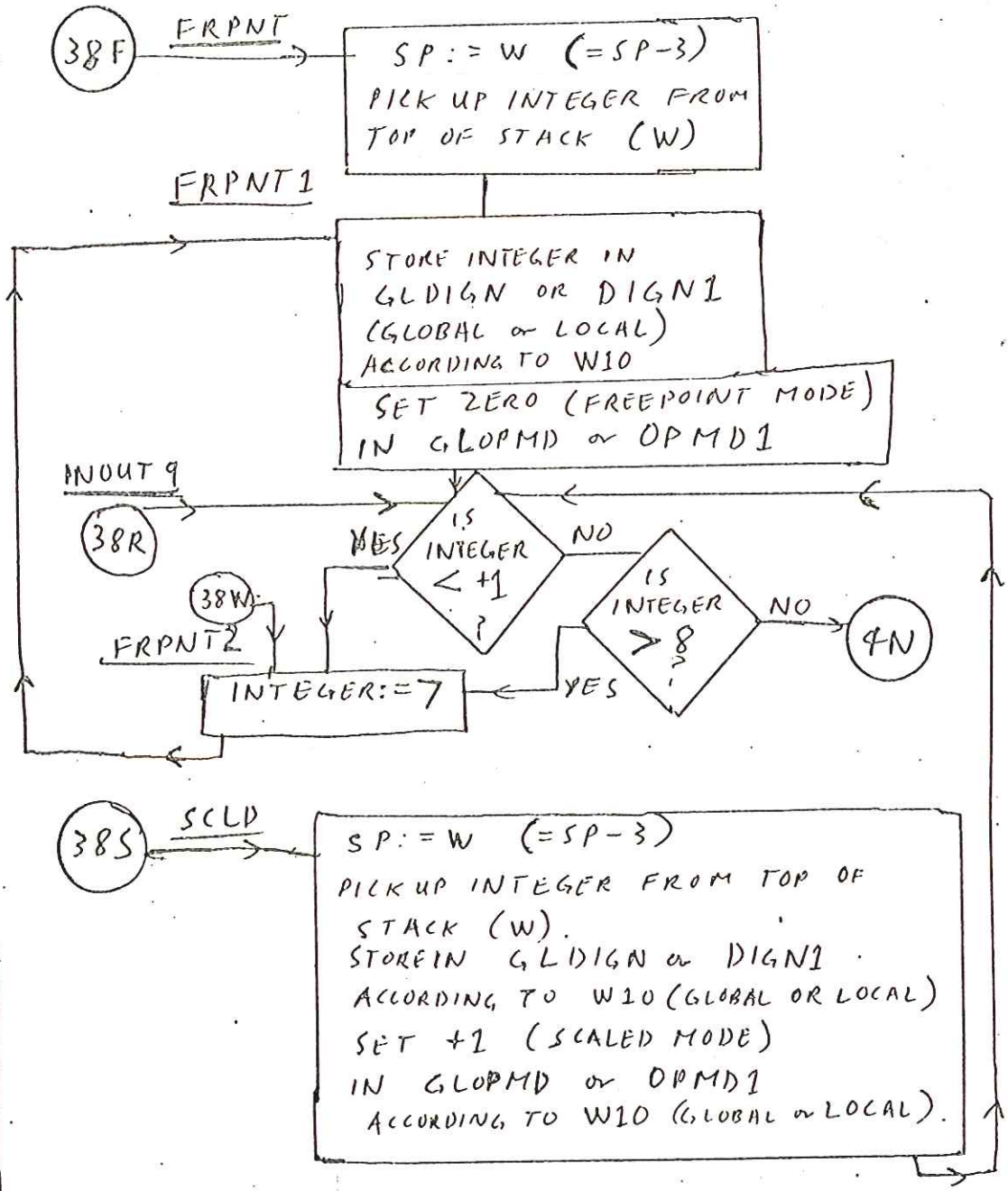

33N
Prim NEGI
INTEGER=CONTENT OF SP-3
IS INTEGER -1*2^17 ?
YES
NEGIE
10E (NEGIE)
NO
CONTENT OF SP-3 := -INTEGER
4N (NXPORD)
Prim RTOI1
33C
SUBROUTINE W3+SR
LOAD CONTENT OF SP-3, SP-2, SP-1 TO WORKSPACE.
SUBROUTINE RICONV (Sheet 10)
STORE INTEGER IN SP-3
4N
Prim ITOR1
33A
W:= SP-3
Prim ITOR2
33B
ITOR2
W:= SP-6
PICK UP INTEGER FROM (W)
SUBROUTINE ITRSB1
CONVERT TO REAL AND STANDARDISE
RRES: STORE REAL NUMBER AT (W).
4N

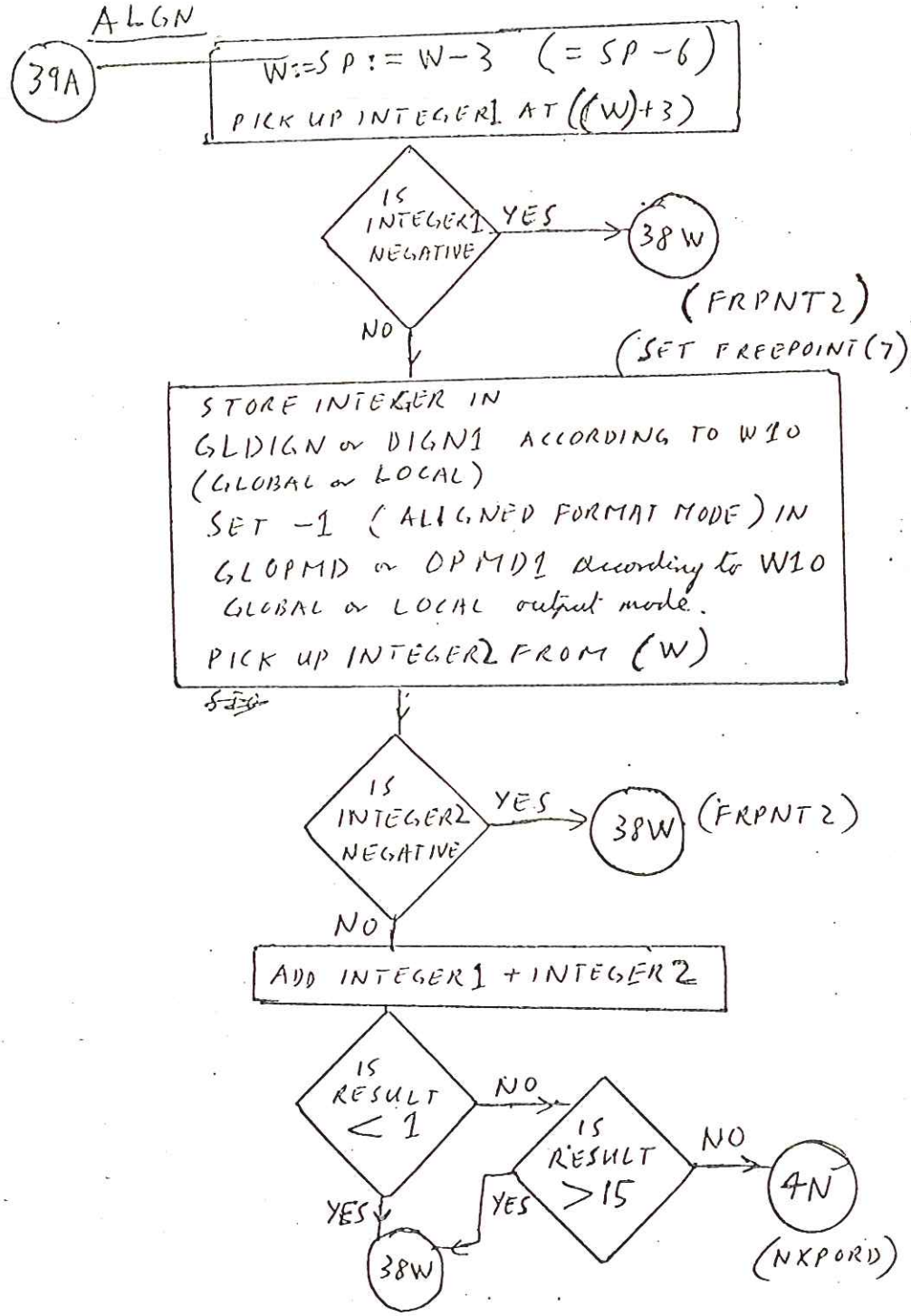

34I
Prim CHECK I
SUBROUTINE CHKSBR
SET DIGITS (6) AND PRINT INTEGER FROM TOP OF STACK. (OUTI5)
RESTORE ORIGONAL PREFIX ADDRESS.
4N (NXPORD)
ENTRY TO CHKSBR
SET IN SAFE STORE CURRENT PREFIX ADDRESS
SET PREFIX TO 'NEWLINE *'
34C
SET OUTPUT DEVICE TO CURRENT GLOBAL OUTPUT DEVICE.
COPY VALUES FROM TOP OF STACK TO WORKSPACE.
EXIT from SUBROUTINE
34R
Prim CHECK R
SUBROUTINE CHKSBR
SET SCALED (8) FORMAT PRINT REAL NUMBER FROM TOP OF STACK (OUTR+3).
34S
Prim CHECKS
SUBROUTINE ENTRY 34C
PRINT STRING "L'*'.
SP:=SP-3;
PRINT STRING GIVEN BY ADDRESS FROM TOP OF STACK.
4N
34B
Prim CHECKB
SUBROUTINE ENTRY 34C
IS 'TOP OF STACK' ZERO
YES
NO
PRINT STRING "L'*'FALSE"'
PRINT STRING "L'*"TRUE"'.
4N

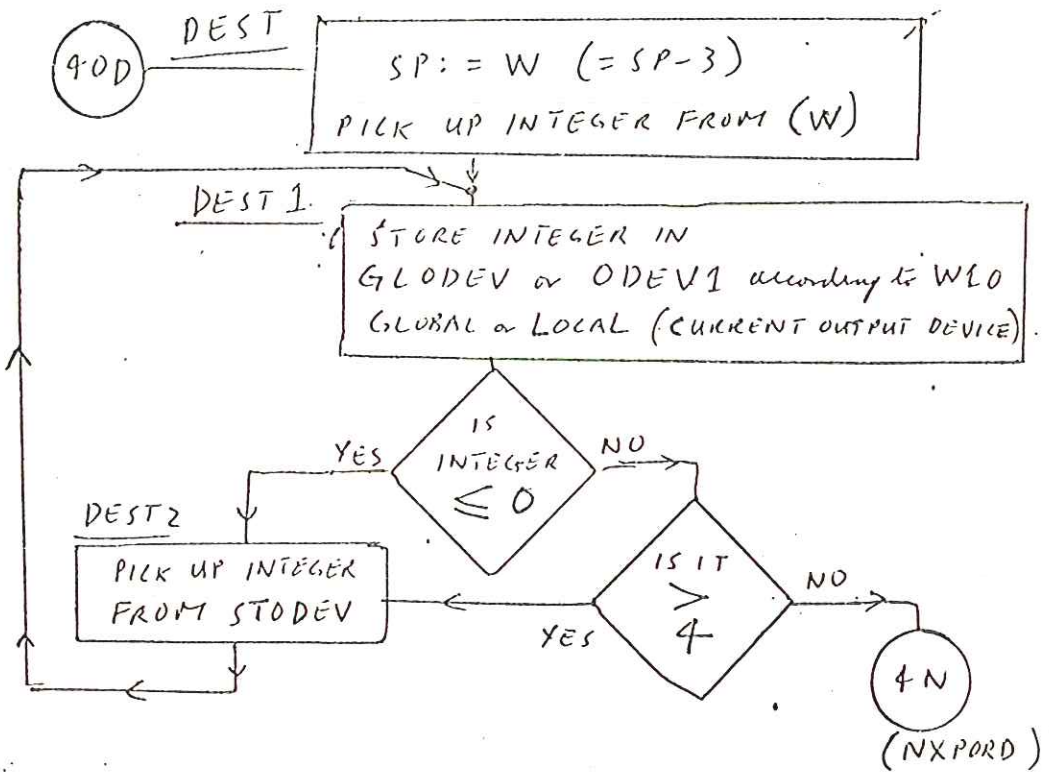

35F
Prim FINISH
OUTPUT 20 BLANKS ON PUNCH
OUTPUT DEVICE (ODEV) :=
CURRENT GLOBAL OUTPUT DEVICE
(GLODEV)
PRINT STRING "L2' FINISH "L2"
SUBROUTINE FINSUB
OUTPUT 150 BLANKS TO PUNCH
35S
STOP
SET CONTINUE ADDRESS
TO (STOP)
7W
(SWAIT)
35N
Prim NEGR
W:=SP-3 (ADDRESS OF REAL)
SUBROUTINE NEGR1
(Sheet 11)
4N
(NXPORD)
35W
Prim WAIT
SET CONTINUE ADDRESS IN PAUSRT
TO GOTO NXPORD (Sheet 4)
(SWAIT)
7W

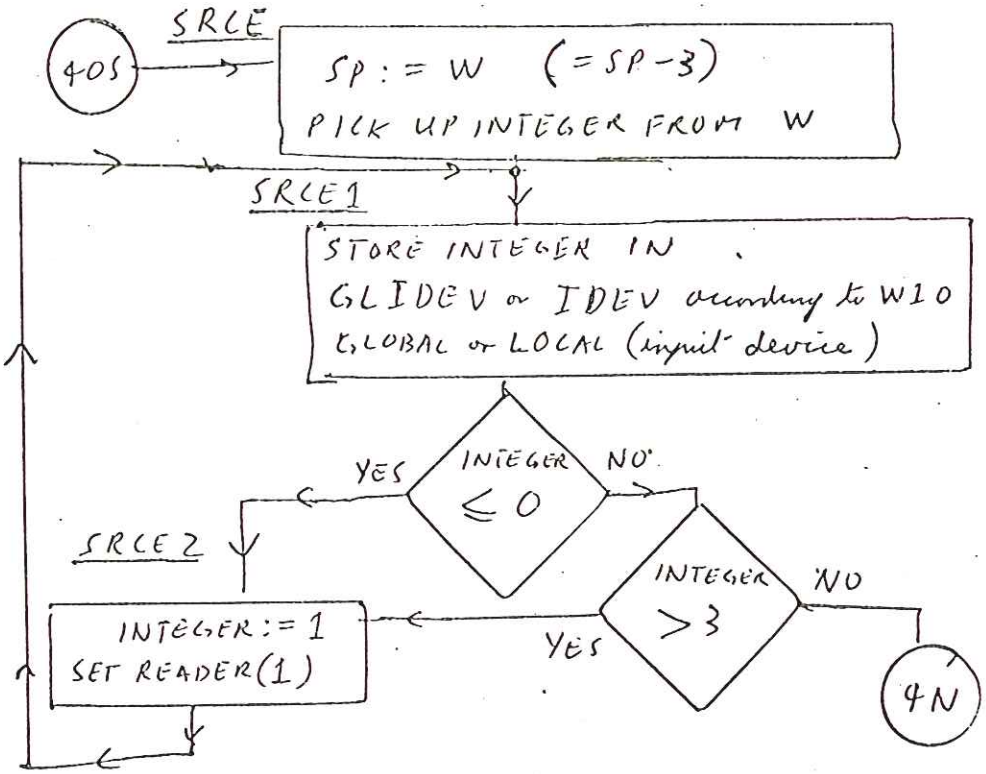

361
Fun INOUT
IS ADPART >15
YES
NO
INOUT1
SET W10:=+7 (LOCAL VALUES)
SET W10:=+0 (GLOBAL VALUES)
INOUT2
W:=SP-3
CALL APPROPRIATE INOUT SUBROUTINE ACCORDING TO VALUE OF ADPART.
0 (NXPORD)
1 RDNM
2 RDNM
3 OUTI
4 OUTR
5 ALGN
6 DEST
7 DIGIT
8 FRPNT
9 SPARE
10 SPARE
11 PREX
12 SMLN
13 SCLD
14 SRCE
15 OSTR1
16 ALGN
17 DEST
18 DIGIT
19 FRPNT
20 INOUT4
21 SPARE
22 PREX
23 SMLN
24 SCLD
25 SRCE
39A
40D
41D
38F
365
37X
37N
385
405
37S
41C
4N
(NXPORD)
ERROR NUMBER := 20 (COMPILER ERROR)
7E


375
OSTR1
SP := W (= SP-3)
OUTPUT DEVICE (ODEV)
:= LOCAL DEVICE (ODEV1)
PICK UP ADDRESS FROM TOP OF STACK. (SP).
PRINT THE STRING STARTING AT THIS ADDRESS (OSTSB Sheet )
4N
(NXPORD)
37L
CHNGOP
STODEV := 4
(STANDARD OUTPUT DEVICE SET TO LINE PRINTER).
37P
PUNCH1
STODEV := 1
(SET PUNCH AS STANDARD OUTPUT DEVICE).
DYNAMIC STOP
37N
SMLN
PICK UP ADDRESS OF NULL STRING ("")
37X
PRFX
SP := W (SP-3)
PICK UP ADDRESS FROM (SP)
(TOP OF STACK)
PRFX1
STORE ADDRESS IN GLPRFA OR PRFA1 (GLOBAL OR LOCAL PREFIX ADDRESS) ACCORING TO VALUE OF W10
(NXPORD)
4N

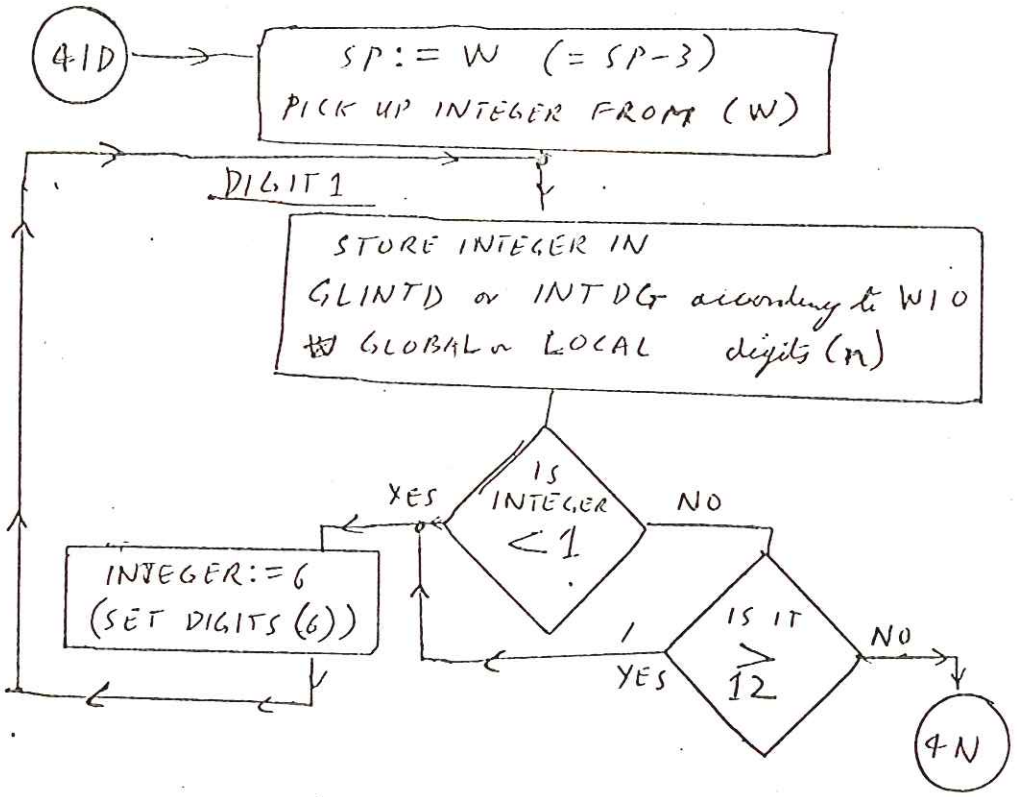

38F
FRPNT
SP := W (=SP-3)
PICK UP INTEGER FROM TOP OF STACK (W)
FRPNT1
STORE INTEGER IN GLDIGN OR DIGN1 (GLOBAL or LOCAL) ACCORDING TO W10
SET ZERO (FREEPOINT MODE) IN GLOPMD or OPMD1
INOUT 9
38R
IS INTEGER < +1 ?
YES
NO
IS INTEGER > 8 ?
NO
9N
38W
FRPNT2
INTEGER:=7
YES
38S
SCLD
SP := W (=SP-3)
PICK UP INTEGER FROM TOP OF STACK (W).
STORE IN GLDIGN or DIGN1 ACCORDING TO W10 (GLOBAL OR LOCAL)
SET +1 (SCALED MODE)
IN GLOPMD or OPMD1 ACCORDING TO W10 (GLOBAL or LOCAL).

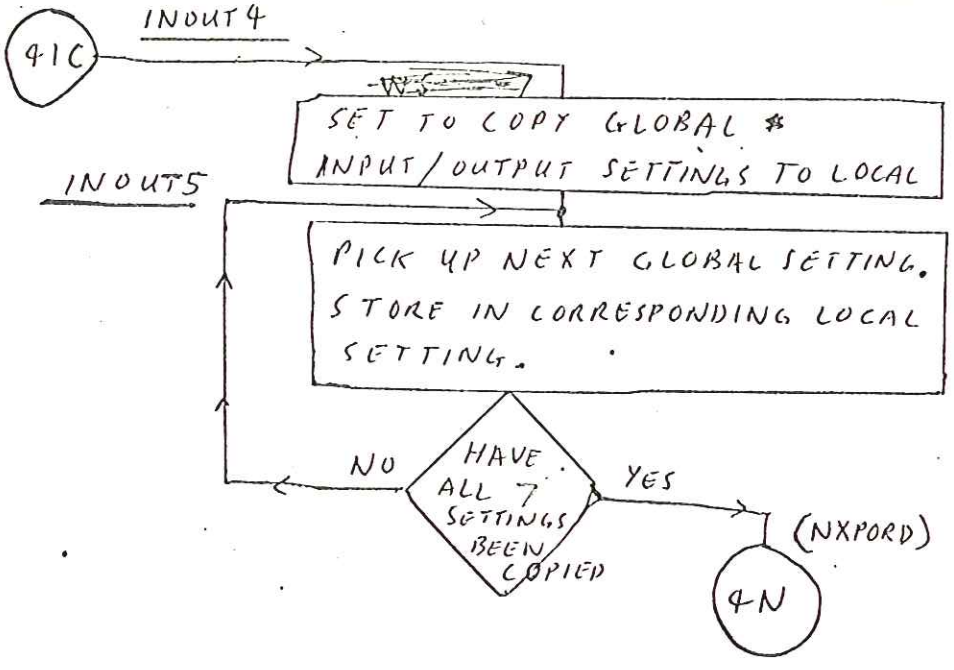

ALGN
39A
W:=SP:=W-3 (=SP-6)
PICK UP INTEGER1 AT ((W)+3)
IS INTEGER1 NEGATIVE
YES
38W
(FRPNT2)
(SET FREEPOINT(7).)
NO
STORE INTEGER IN GLDIGN or DIGN1 ACCORDING TO W10 (GLOBAL or LOCAL)
SET -1 (ALIGNED FORMAT MODE) IN GLOPMD or OPMD1 according to W10 GLOBAL or LOCAL output mode.
PICK UP INTEGER2 FROM (W)
IS INTEGER2 NEGATIVE
YES
38W
(FRPNT2)
NO
ADD INTEGER1 + INTEGER2
IS RESULT < 1
NO
IS RESULT > 15
NO
4N
(NXPORD)
YES
YES
38W


40D
DEST
SP := W (= SP-3)
PICK UP INTEGER FROM (W)
DEST 1
STORE INTEGER IN
GLODEV or ODEV1 according to W10
GLOBAL or LOCAL (CURRENT OUTPUT DEVICE)
IS INTEGER ≤ 0
YES
NO
DEST 2
PICK UP INTEGER FROM STODEV
IS IT > 4
YES
NO
4N
(NXPORD)

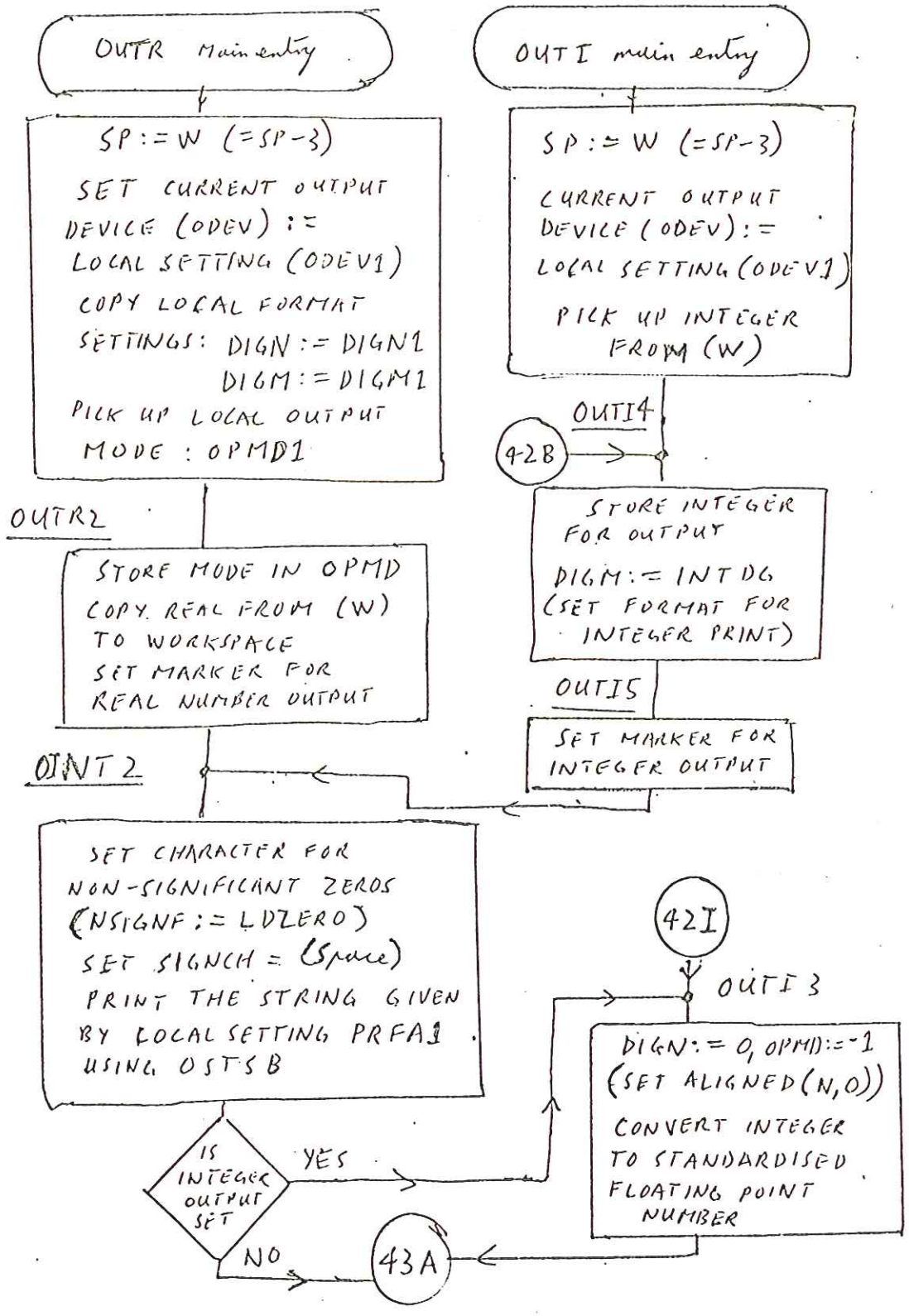

40S
SRCE
SP := W (= SP-3)
PICK UP INTEGER FROM W
SRCE 1
STORE INTEGER IN
GLIDEV or IDEV according to W10
GLOBAL or LOCAL (input device)
INTEGER ≤ 0
YES
NO
SRCE 2
INTEGER := 1
SET READER(1)
INTEGER > 3
YES
NO
4N

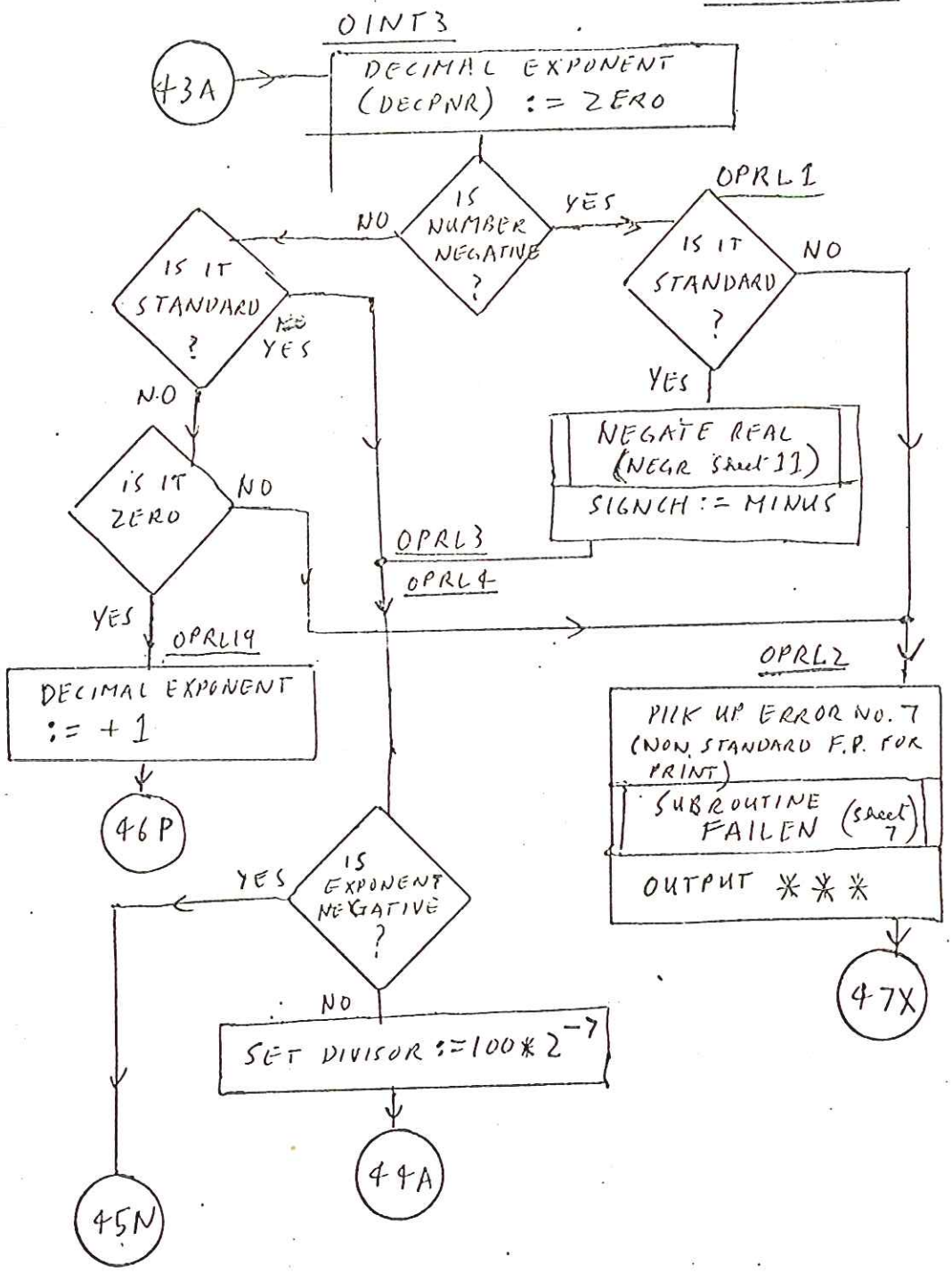
DIGIT

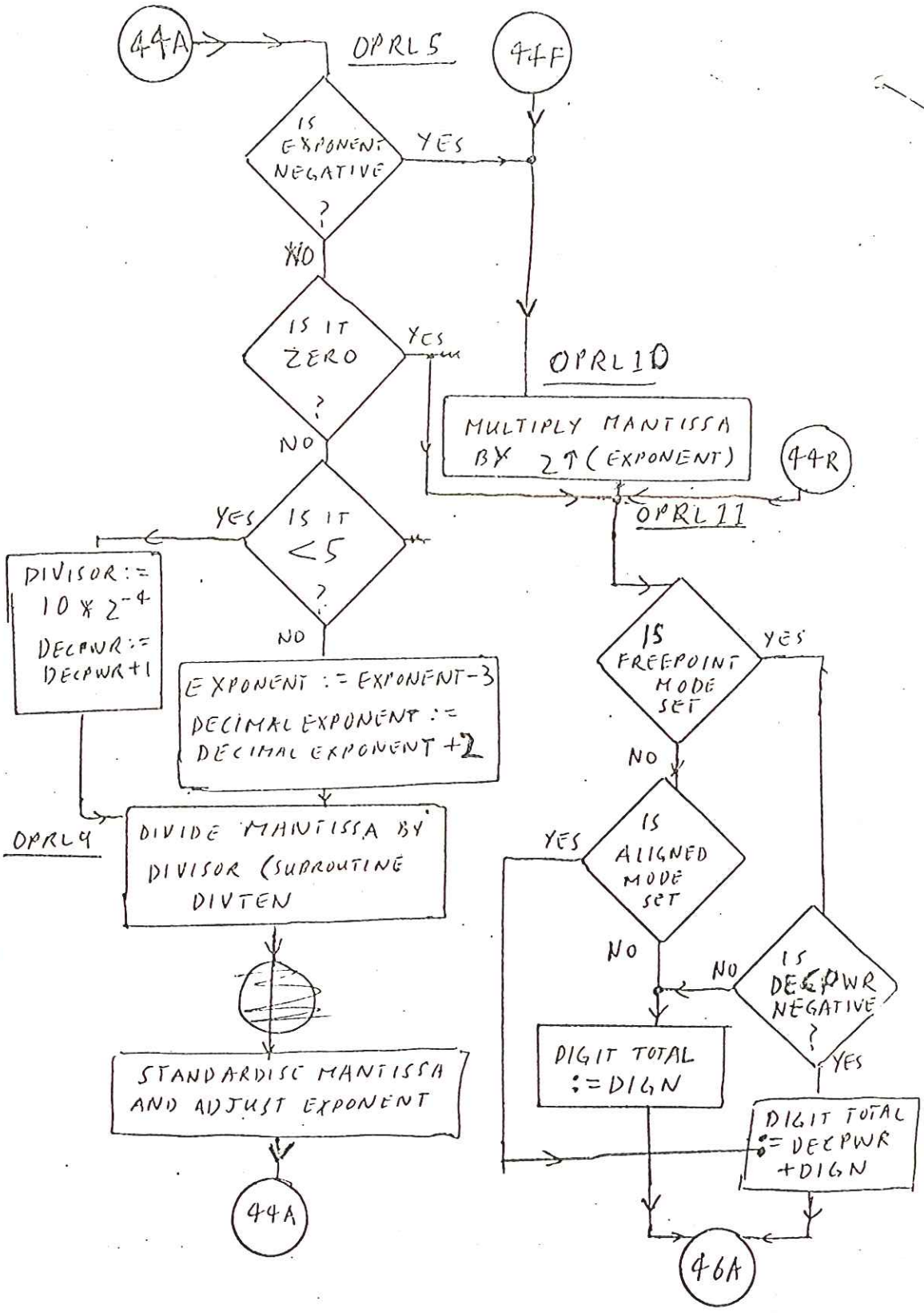

41D
SP := W (= SP-3)
PICK UP INTEGER FROM (W)
DIGIT 1
STORE INTEGER IN
GLINTD or INTDG according to W10
GLOBAL or LOCAL digits (n)
IS INTEGER < 1
YES
NO
INTEGER := 6
(SET DIGITS (6))
IS IT > 12
YES
NO
4N

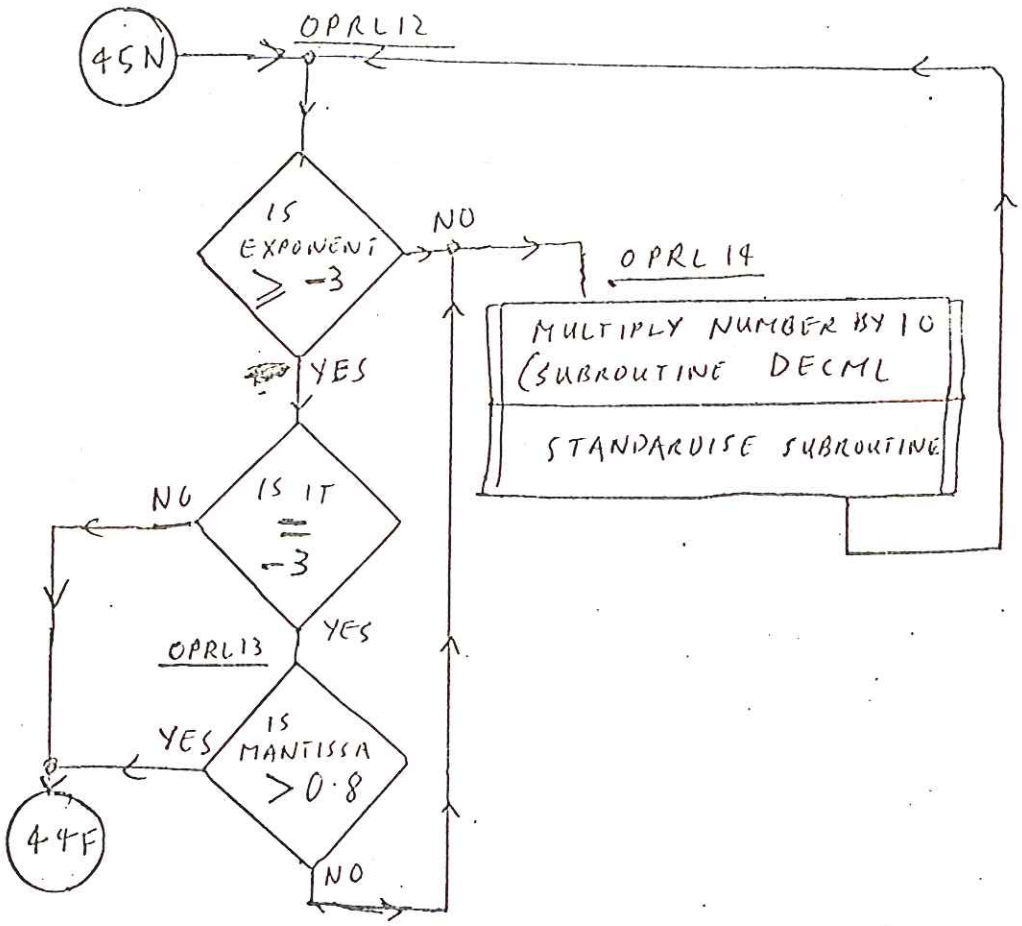

41C
INOUT4
SET TO COPY GLOBAL INPUT/OUTPUT SETTINGS TO LOCAL
INOUT5
PICK UP NEXT GLOBAL SETTING. STORE IN CORRESPONDING LOCAL SETTING.
HAVE ALL 7 SETTINGS BEEN COPIED
NO
YES
(NXPORD)
4N

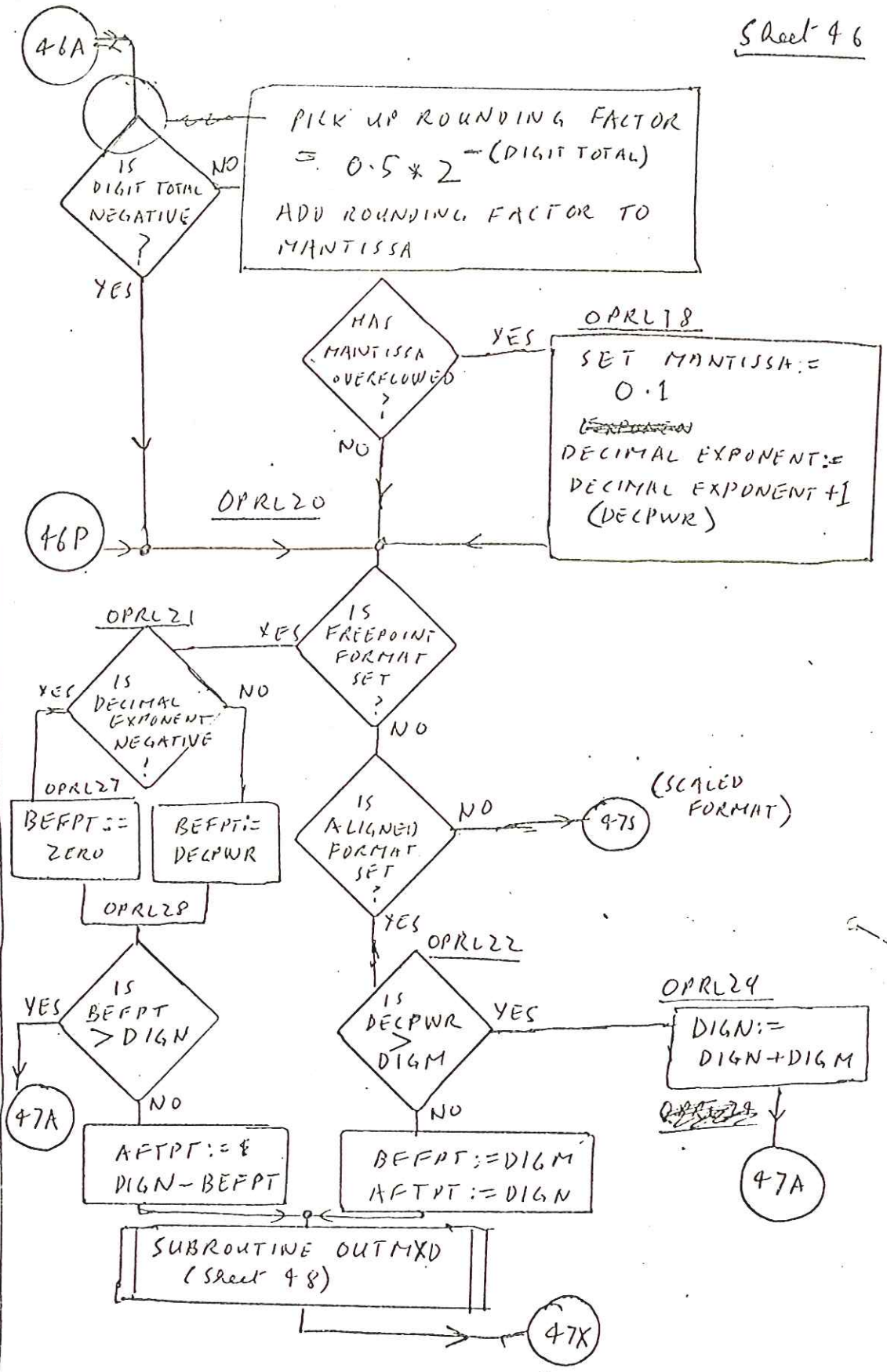
# OUTPUT SUBROUTINE

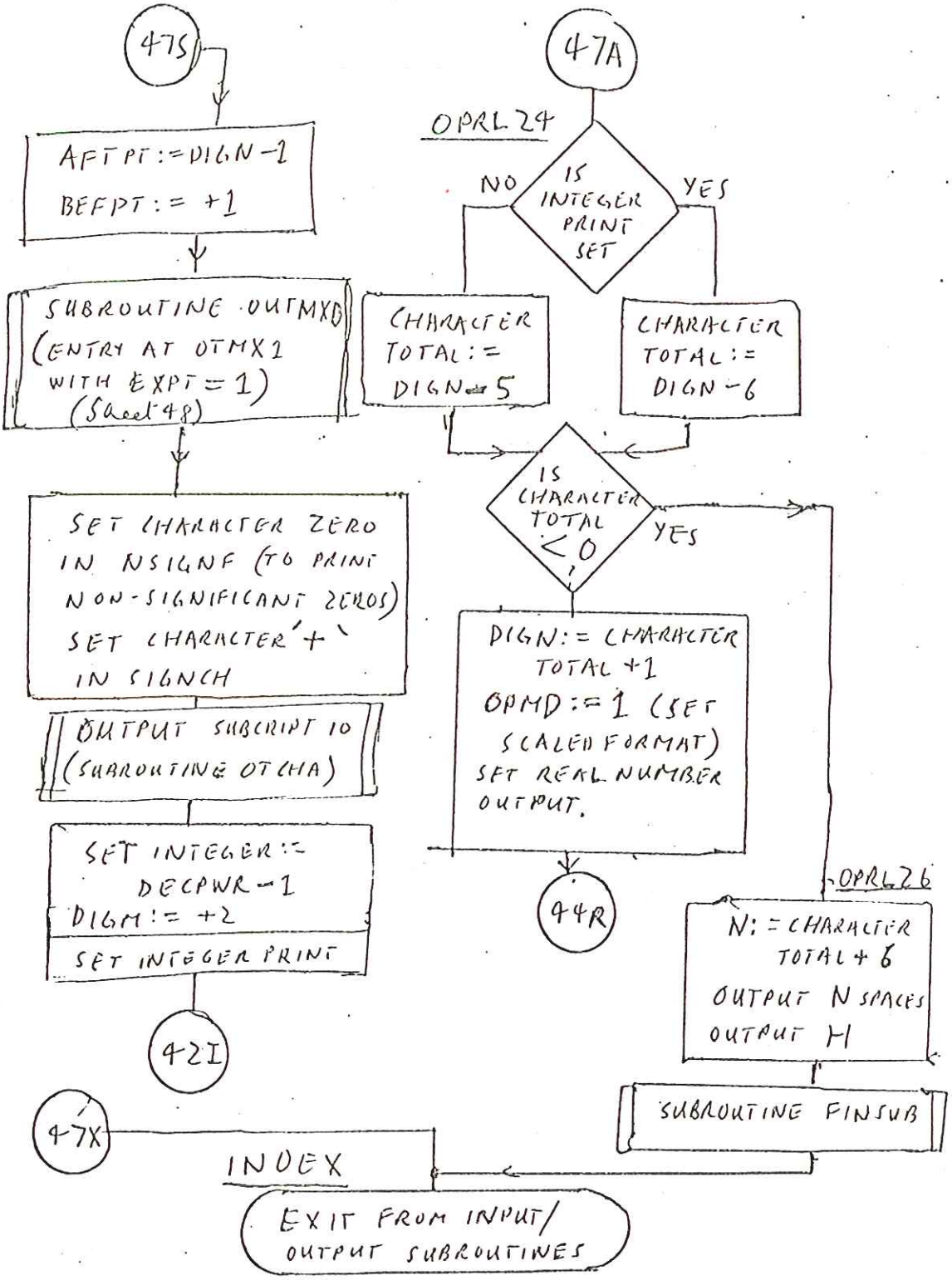


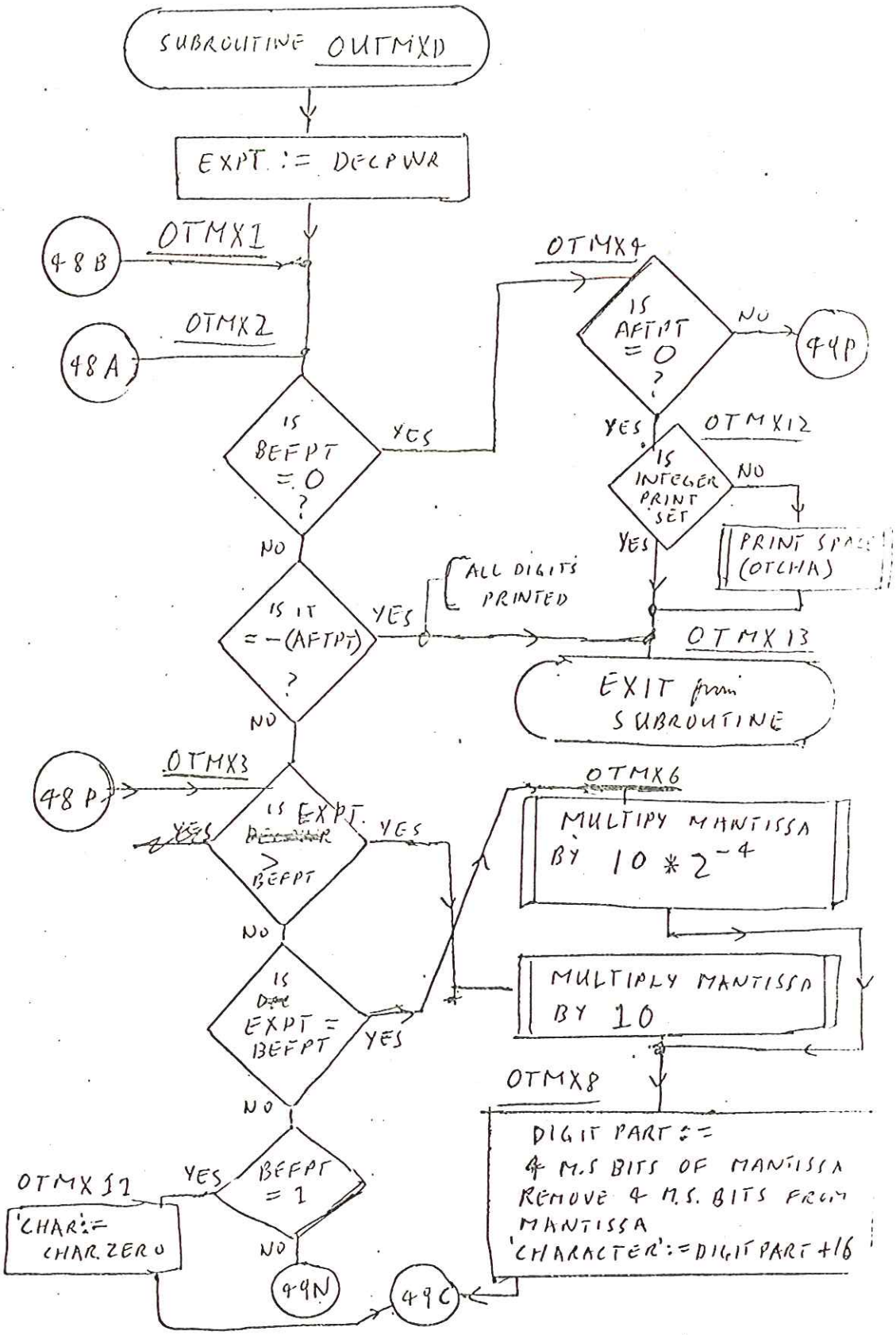

OINT3
43A
DECIMAL EXPONENT (DECPNR) := ZERO
IS NUMBER NEGATIVE ?
NO
YES
OPRL1
IS IT STANDARD ?
NO
YES
IS IT STANDARD ?
NO
YES
NEGATE REAL (NEGR Sheet 11)
SIGNCH := MINUS
IS IT ZERO
NO
YES
OPRL3
OPRL4
OPRL19
DECIMAL EXPONENT := +1
OPRL2
PICK UP ERROR NO. 7 (NON STANDARD F.P. FOR PRINT)
SUBROUTINE FAILEN (sheet 7)
OUTPUT ***
46P
47X
IS EXPONENT NEGATIVE ?
YES
NO
SET DIVISOR := 100 * 2^-7
44A
45N

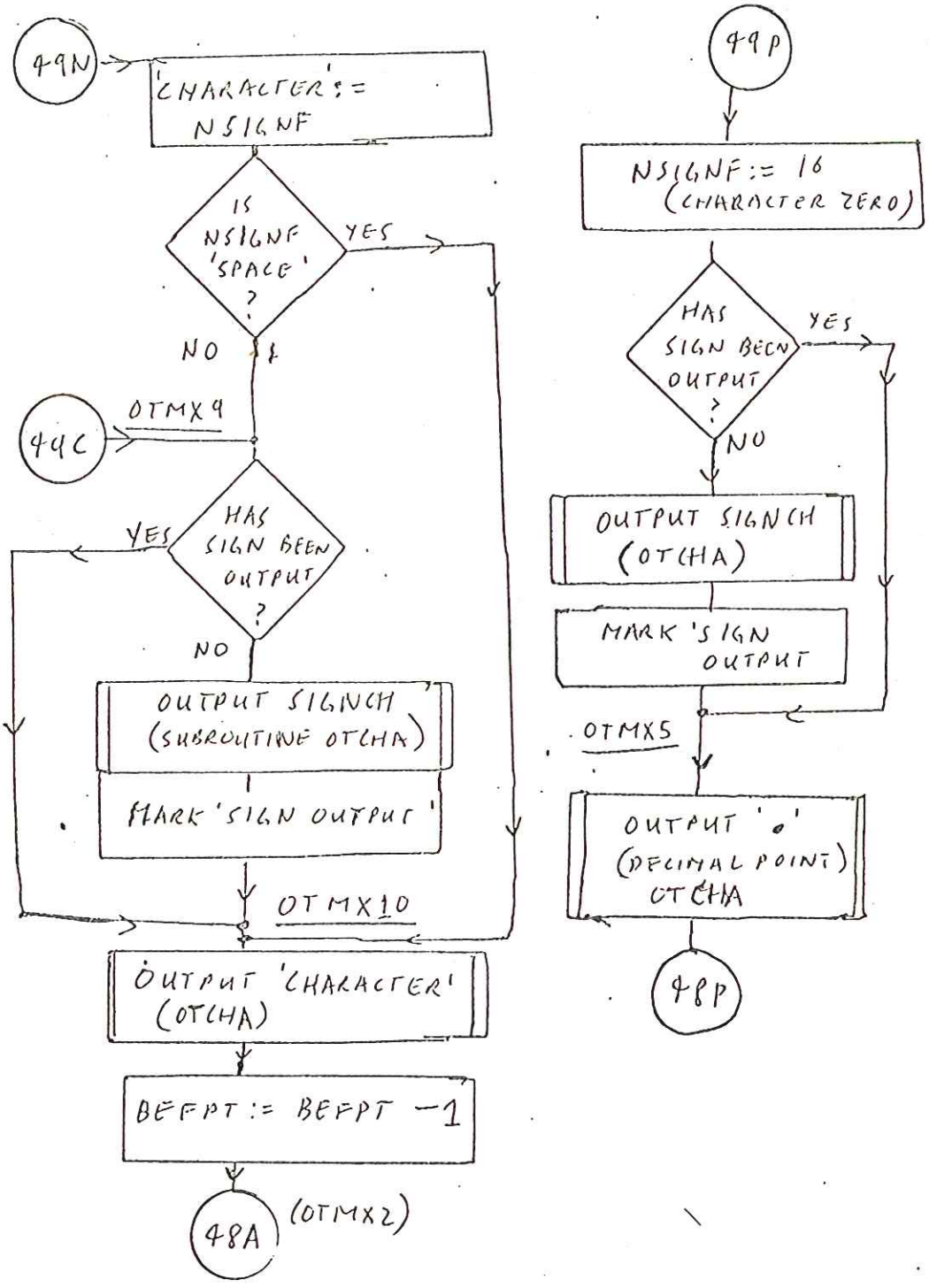

44A
OPRL5
44F
IS EXPONENT NEGATIVE ?
YES
NO
IS IT ZERO ?
YES
NO
OPRL10
MULTIPLY MANTISSA BY 2↑(EXPONENT)
44R
OPRL11
IS IT < 5 ?
YES
NO
DIVISOR := 10 * 2^-4
DECPWR := DECPWR+1
EXPONENT := EXPONENT-3
DECIMAL EXPONENT := DECIMAL EXPONENT +2
OPRL9
DIVIDE MANTISSA BY DIVISOR (SUBROUTINE DIVTEN
STANDARDISE MANTISSA AND ADJUST EXPONENT
44A
IS FREEPOINT MODE SET
YES
NO
IS ALIGNED MODE SET
YES
NO
IS DECPWR NEGATIVE ?
NO
YES
DIGIT TOTAL := DIGN
DIGIT TOTAL := DECPWR +DIGN
46A


45N
OPRL12
IS EXPONENT ≥ -3
NO
OPRL 14
MULTIPLY NUMBER BY 10 (SUBROUTINE DECML
STANDARDISE SUBROUTINE
YES
NO
IS IT = -3
YES
OPRL13
YES
IS MANTISSA > 0·8
NO
44F

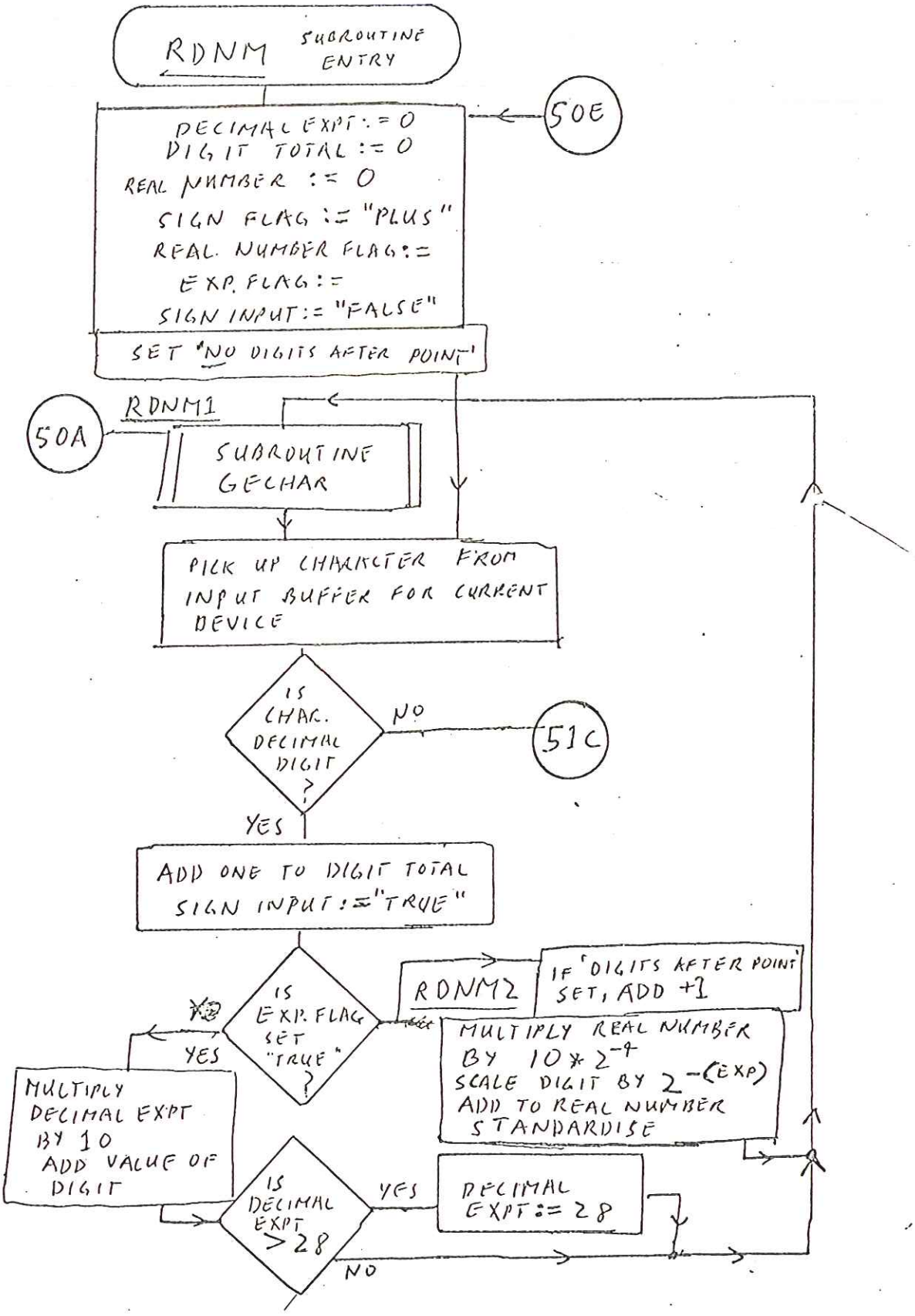

46A
IS DIGIT TOTAL NEGATIVE ?
NO
YES
PICK UP ROUNDING FACTOR = 0.5 * 2 ** -(DIGIT TOTAL)
ADD ROUNDING FACTOR TO MANTISSA
HAS MANTISSA OVERFLOWED ?
YES
NO
OPRL18
SET MANTISSA := 0.1
DECIMAL EXPONENT := DECIMAL EXPONENT + 1 (DECPWR)
46P
OPRL20
IS FREEPOINT FORMAT SET ?
YES
NO
OPRL21
IS DECIMAL EXPONENT NEGATIVE ?
YES
NO
OPRL27
BEFPT := ZERO
BEFPT := DECPWR
OPRL28
IS ALIGNED FORMAT SET ?
NO
(SCALED FORMAT)
47S
YES
OPRL22
IS BEFPT > DIGN
YES
NO
47A
AFTPT := DIGN - BEFPT
IS DECPWR > DIGM
YES
NO
OPRL24
DIGN := DIGN + DIGM
47A
BEFPT := DIGM
AFTPT := DIGN
SUBROUTINE OUTMXD (Sheet 48)
47X

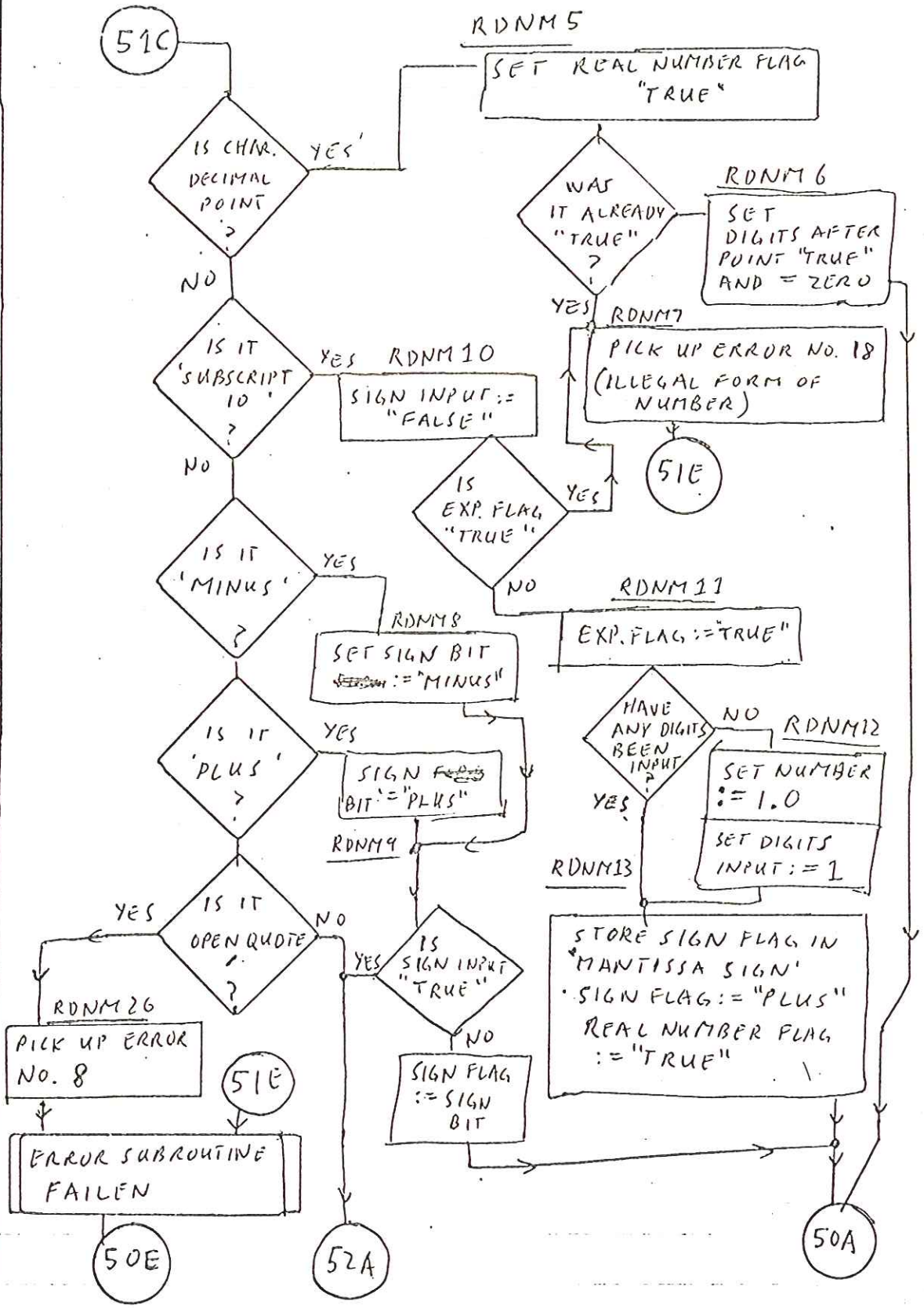

47S
AFTPT := DIGN - 1
BEFPT := +1
SUBROUTINE OUTMXD (ENTRY AT OTMX1 WITH EXPT = 1) (Sheet 48)
SET CHARACTER ZERO IN NSIGNF (TO PRINT NON-SIGNIFICANT ZEROS) SET CHARACTER '+' IN SIGNCH
OUTPUT SUBCRIPT 10 (SUBROUTINE OTCHA)
SET INTEGER := DECPWR - 1
DIGM := +2
SET INTEGER PRINT
42I
47X
INDEX
EXIT FROM INPUT/ OUTPUT SUBROUTINES
47A
OPRL24
IS INTEGER PRINT SET
NO
YES
CHARACTER TOTAL := DIGN + 5
CHARACTER TOTAL := DIGN - 6
IS CHARACTER TOTAL < 0
YES
DIGN := CHARACTER TOTAL + 1
OPMD := 1 (SET SCALED FORMAT)
SET REAL NUMBER OUTPUT.
44R
OPRL26
N := CHARACTER TOTAL + 6
OUTPUT N SPACES
OUTPUT H
SUBROUTINE FINSUB

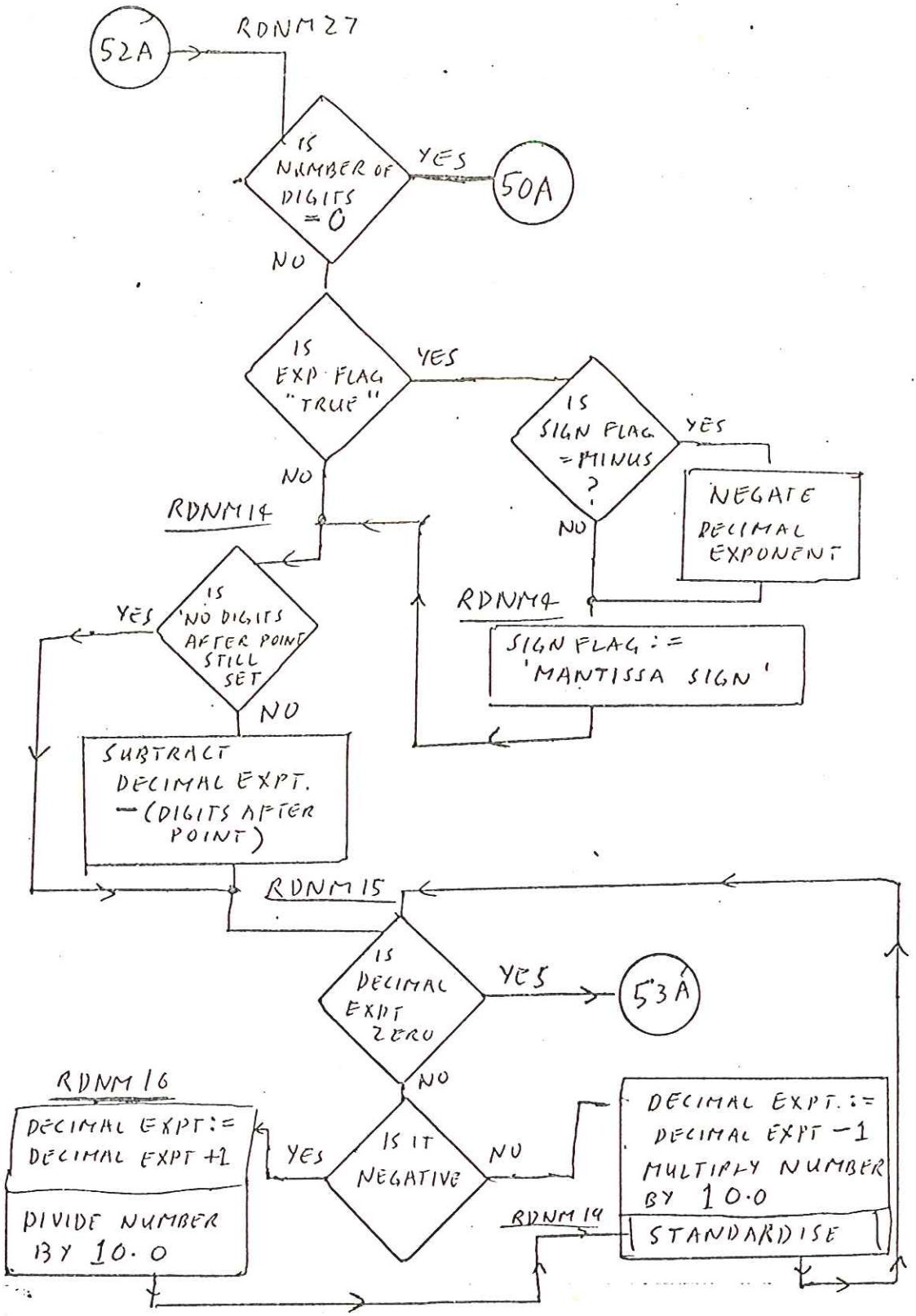

SUBROUTINE OUTMXD
EXPT := DECPWR
OTMX1
48B
OTMX2
48A
IS BEFPT = 0 ?
YES
NO
OTMX4
IS AFTPT = 0 ?
NO
49P
YES
OTMX12
IS INTEGER PRINT SET
NO
PRINT SPACE (OTCHR)
YES
ALL DIGITS PRINTED
IS IT = -(AFTPT) ?
YES
NO
OTMX13
EXIT from SUBROUTINE
OTMX3
48P
IS EXPT > BEFPT
YES
NO
OTMX6
MULTIPY MANTISSA BY 10 * 2^-4
IS EXPT = BEFPT
YES
NO
MULTIPLY MANTISSA BY 10
OTMX8
DIGIT PART := 4 M.S BITS OF MANTISSA
REMOVE 4 M.S. BITS FROM MANTISSA
'CHARACTER' := DIGIT PART + 16
BEFPT = 1
YES
NO
OTMX11
'CHAR' := CHAR.ZERO
49N
49C

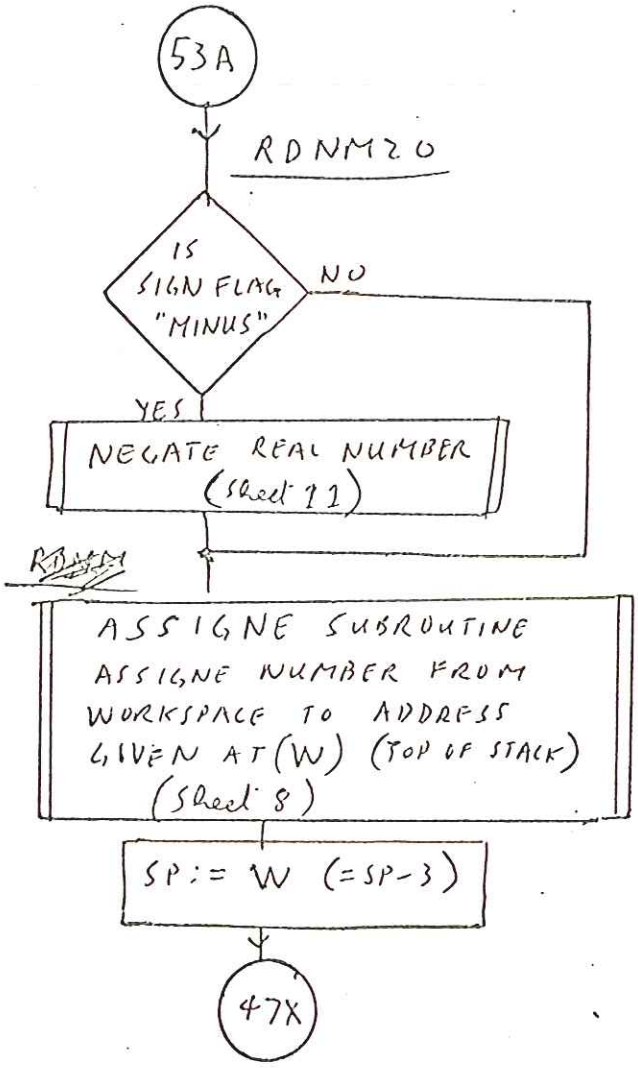

49N
'CHARACTER' := NSIGNF
IS NSIGNF 'SPACE' ?
YES
NO
49C
OTMX9
HAS SIGN BEEN OUTPUT ?
YES
NO
OUTPUT SIGNCH (SUBROUTINE OTCHA)
MARK 'SIGN OUTPUT'
OTMX10
OUTPUT 'CHARACTER' (OTCHA)
BEFPT := BEFPT - 1
48A
(OTMX2)
49P
NSIGNF := 16 (CHARACTER ZERO)
HAS SIGN BEEN OUTPUT ?
YES
NO
OUTPUT SIGNCH (OTCHA)
MARK 'SIGN OUTPUT
OTMX5
OUTPUT '.' (DECIMAL POINT) OTCHA
48P

# INPUT SUBROUTINE

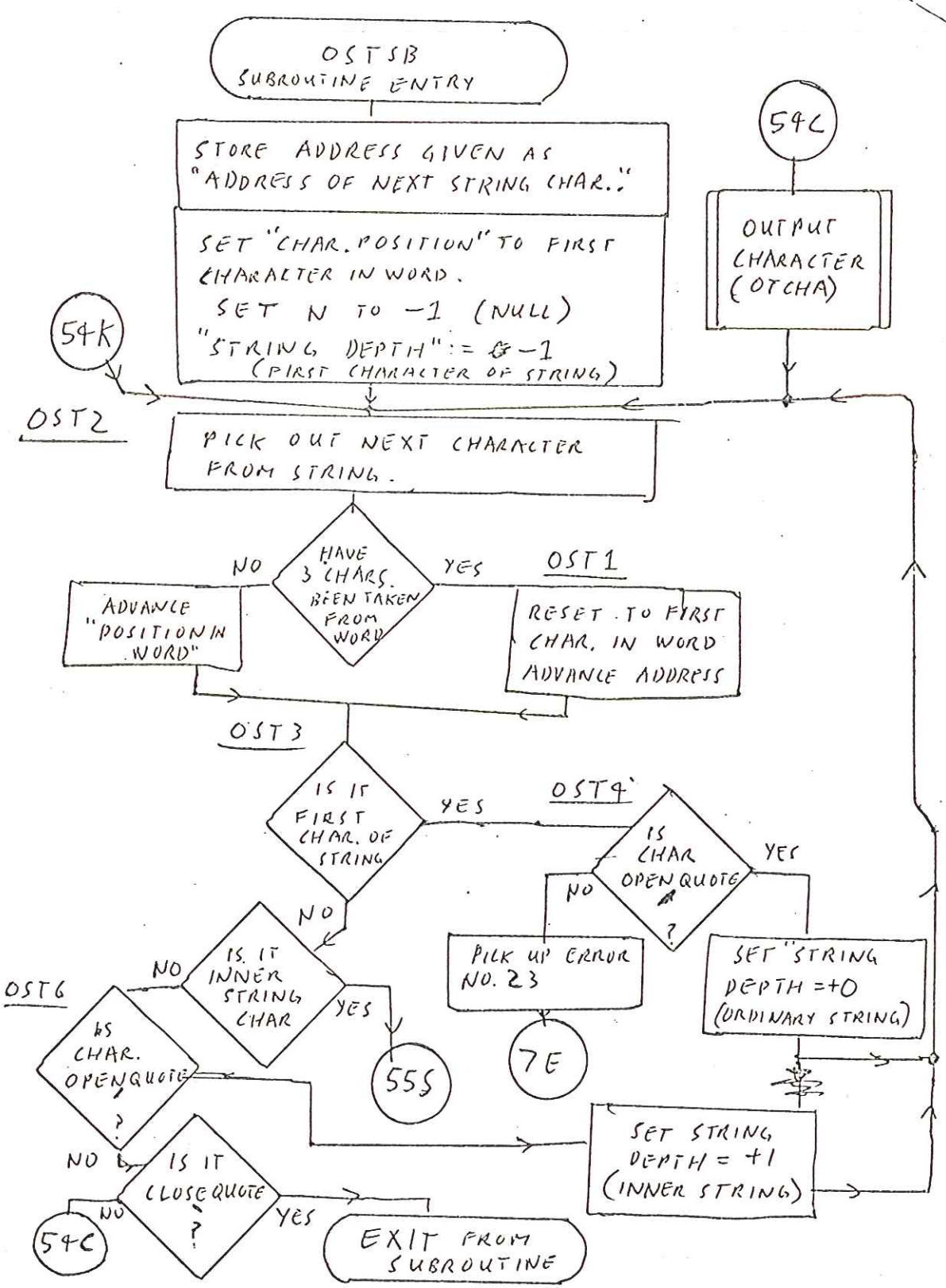


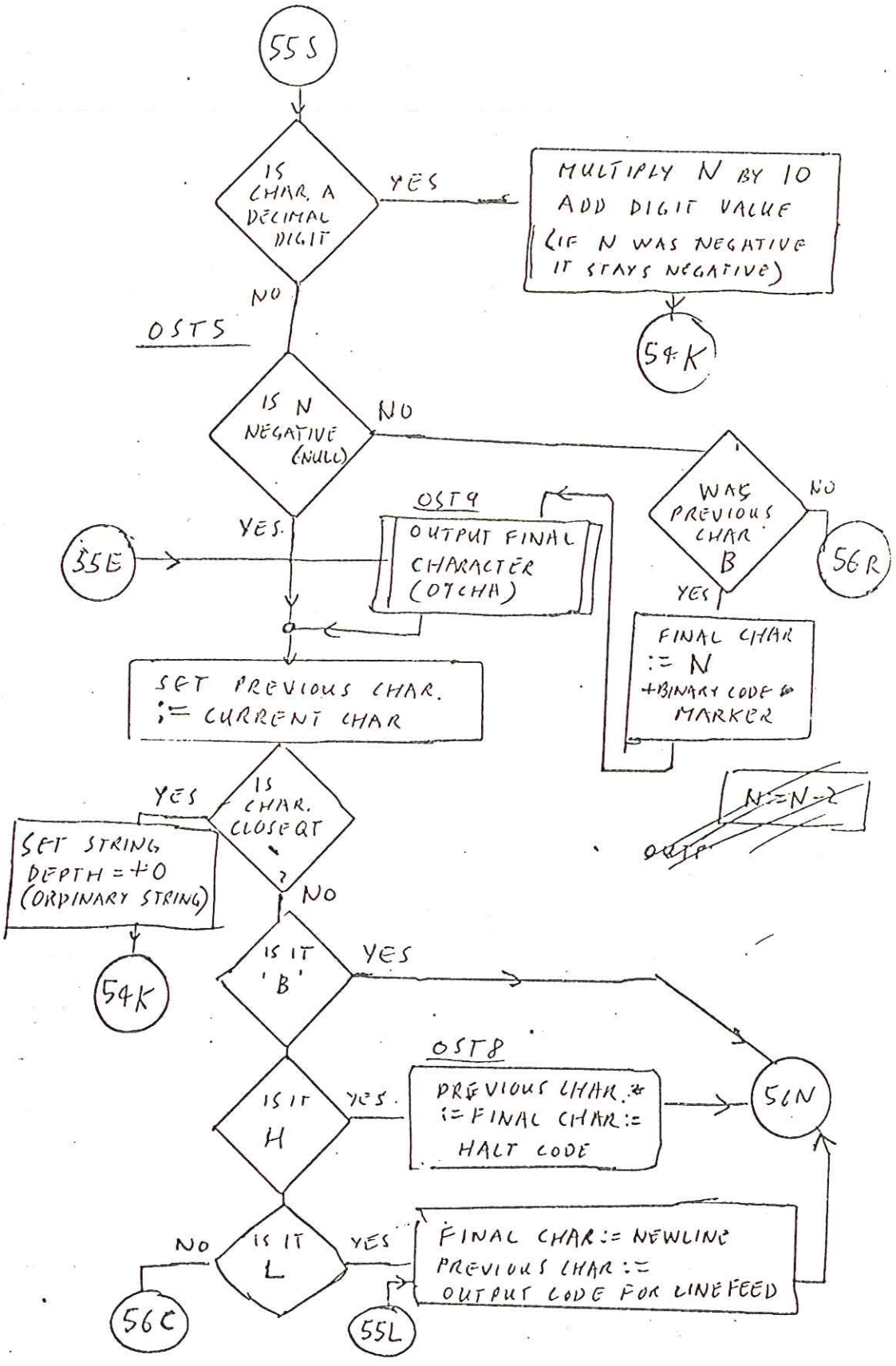

51C
IS CHR. DECIMAL POINT ?
YES
NO
RDNM5
SET REAL NUMBER FLAG "TRUE"
WAS IT ALREADY "TRUE" ?
YES
RDNM6
SET DIGITS AFTER POINT "TRUE" AND = ZERO
IS IT 'SUBSCRIPT 10' ?
YES
RDNM10
SIGN INPUT := "FALSE"
NO
RDNM7
PICK UP ERROR NO. 18 (ILLEGAL FORM OF NUMBER)
51E
IS EXP. FLAG "TRUE"
YES
NO
RDNM11
EXP. FLAG := "TRUE"
IS IT 'MINUS' ?
YES
RDNM8
SET SIGN BIT := "MINUS"
IS IT 'PLUS' ?
YES
SIGN BIT := "PLUS"
RDNM9
HAVE ANY DIGITS BEEN INPUT ?
NO
RDNM12
SET NUMBER := 1.0
SET DIGITS INPUT := 1
YES
RDNM13
IS IT OPEN QUOTE ' ?
YES
NO
RDNM26
PICK UP ERROR No. 8
51E
ERROR SUBROUTINE FAILEN
50E
IS SIGN INPUT "TRUE"
YES
NO
SIGN FLAG := SIGN BIT
STORE SIGN FLAG IN 'MANTISSA SIGN'
SIGN FLAG := "PLUS"
REAL NUMBER FLAG := "TRUE"
52A
50A

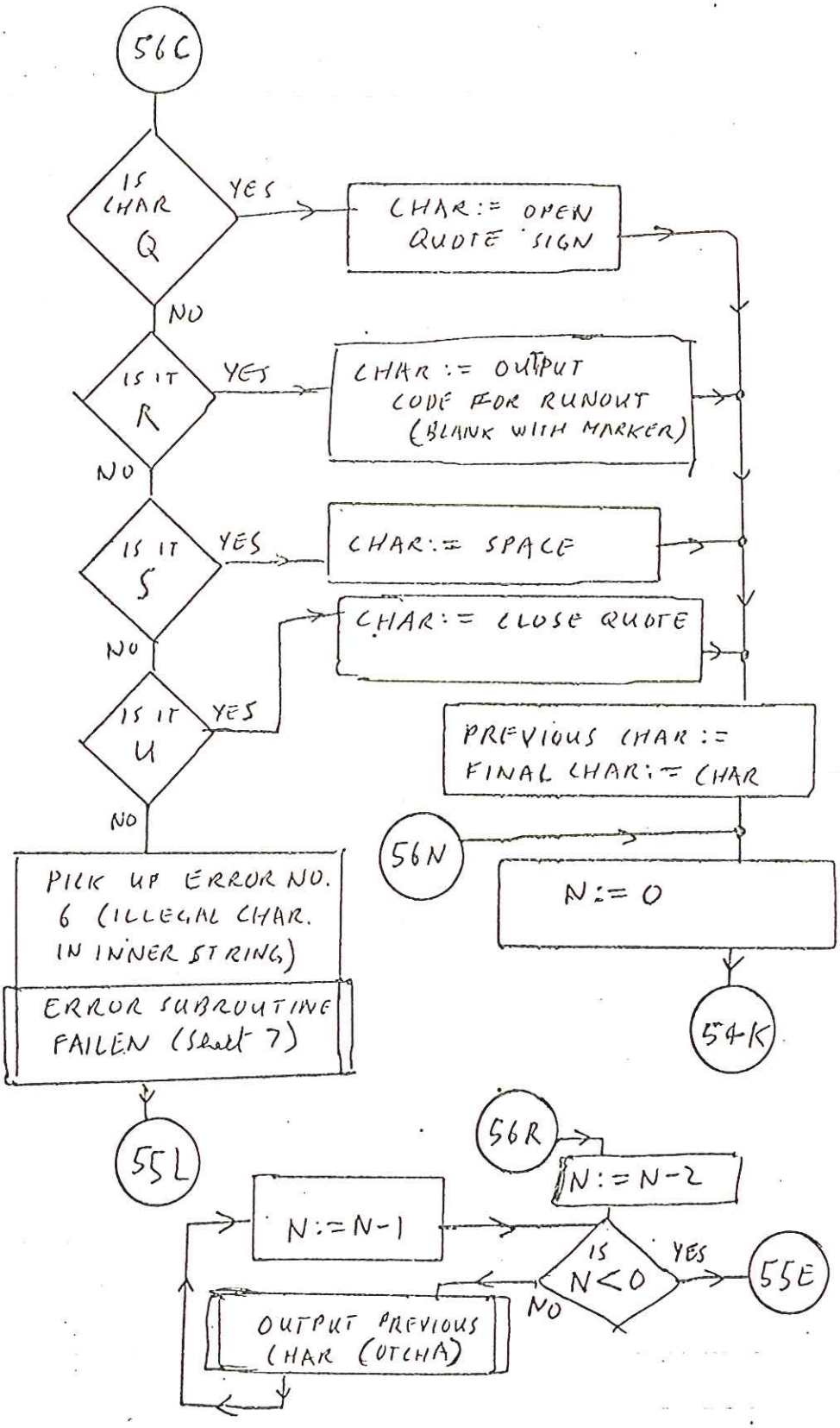

52A
RDNM 27
IS NUMBER OF DIGITS = 0
YES
50A
NO
IS EXP FLAG "TRUE"
YES
IS SIGN FLAG = MINUS ?
YES
NEGATE DECIMAL EXPONENT
NO
NO
RDNM 14
RDNM 4
IS 'NO DIGITS AFTER POINT' STILL SET
YES
NO
SIGN FLAG := 'MANTISSA SIGN'
SUBTRACT DECIMAL EXPT. — (DIGITS AFTER POINT)
RDNM 15
IS DECIMAL EXPT ZERO
YES
53A
NO
RDNM 16
DECIMAL EXPT := DECIMAL EXPT + 1
DIVIDE NUMBER BY 10.0
YES
IS IT NEGATIVE
NO
DECIMAL EXPT. := DECIMAL EXPT — 1
MULTIPLY NUMBER BY 10.0
RDNM 19
STANDARDISE


53A
RDNM20
IS SIGN FLAG "MINUS"
NO
YES
NEGATE REAL NUMBER (Sheet 11)
ASSIGNE SUBROUTINE ASSIGNE NUMBER FROM WORKSPACE TO ADDRESS GIVEN AT (W) (TOP OF STACK) (Sheet 8)
SP := W (=SP-3)
47X

# OUTPUT STRING SUBROUTINE.

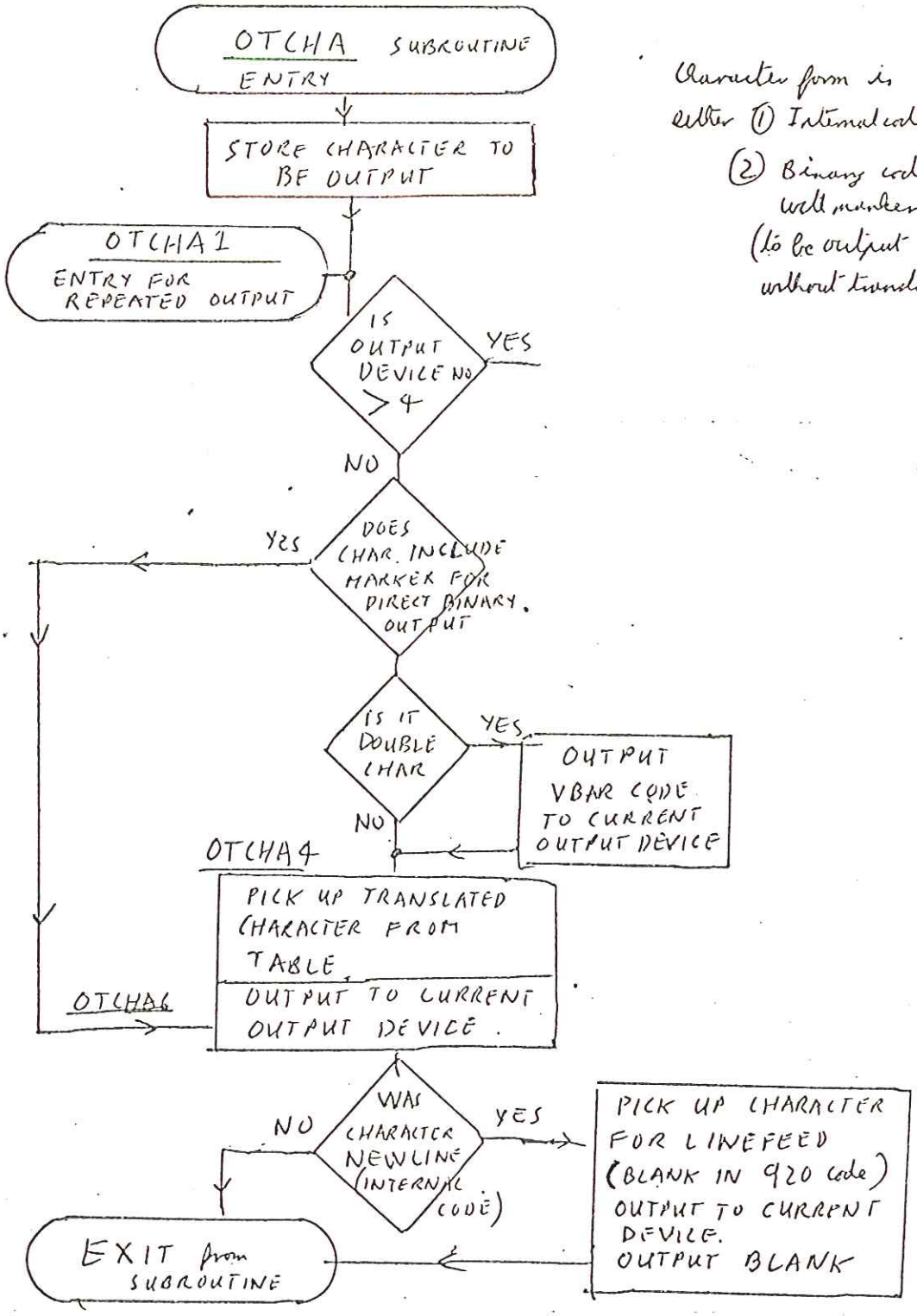



55S
IS CHAR. A DECIMAL DIGIT
YES
MULTIPLY N BY 10
ADD DIGIT VALUE
(IF N WAS NEGATIVE IT STAYS NEGATIVE)
54K
NO
OST5
IS N NEGATIVE (NULL)
NO
WAS PREVIOUS CHAR B
NO
56R
YES
OST9
55E
YES.
OUTPUT FINAL CHARACTER (OTCHA)
FINAL CHAR := N +BINARY CODE & MARKER
N:=N-1
SET PREVIOUS CHAR. := CURRENT CHAR
IS CHAR. CLOSEQT ?
YES
SET STRING DEPTH = +0 (ORDINARY STRING)
54K
NO
IS IT 'B'
YES
OST8
IS IT H
YES
PREVIOUS CHAR. := FINAL CHAR := HALT CODE
56N
IS IT L
NO
YES
FINAL CHAR := NEWLINE
PREVIOUS CHAR := OUTPUT CODE FOR LINEFEED
56C
55L


56C
IS CHAR Q
YES
CHAR := OPEN QUOTE SIGN
NO
IS IT R
YES
CHAR := OUTPUT CODE FOR RUNOUT (BLANK WITH MARKER)
NO
IS IT S
YES
CHAR := SPACE
NO
IS IT U
YES
CHAR := CLOSE QUOTE
NO
PICK UP ERROR NO. 6 (ILLEGAL CHAR. IN INNER STRING)
ERROR SUBROUTINE FAILEN (Sheet 7)
55L
PREVIOUS CHAR := FINAL CHAR := CHAR
56N
N := 0
54K
56R
N := N-2
N := N-1
IS N<0
YES
55E
NO
OUTPUT PREVIOUS CHAR (OTCHA)

# SUBROUTINE : OUTPUT ONE CHARACTER

(Common to 903 and 920 versions)

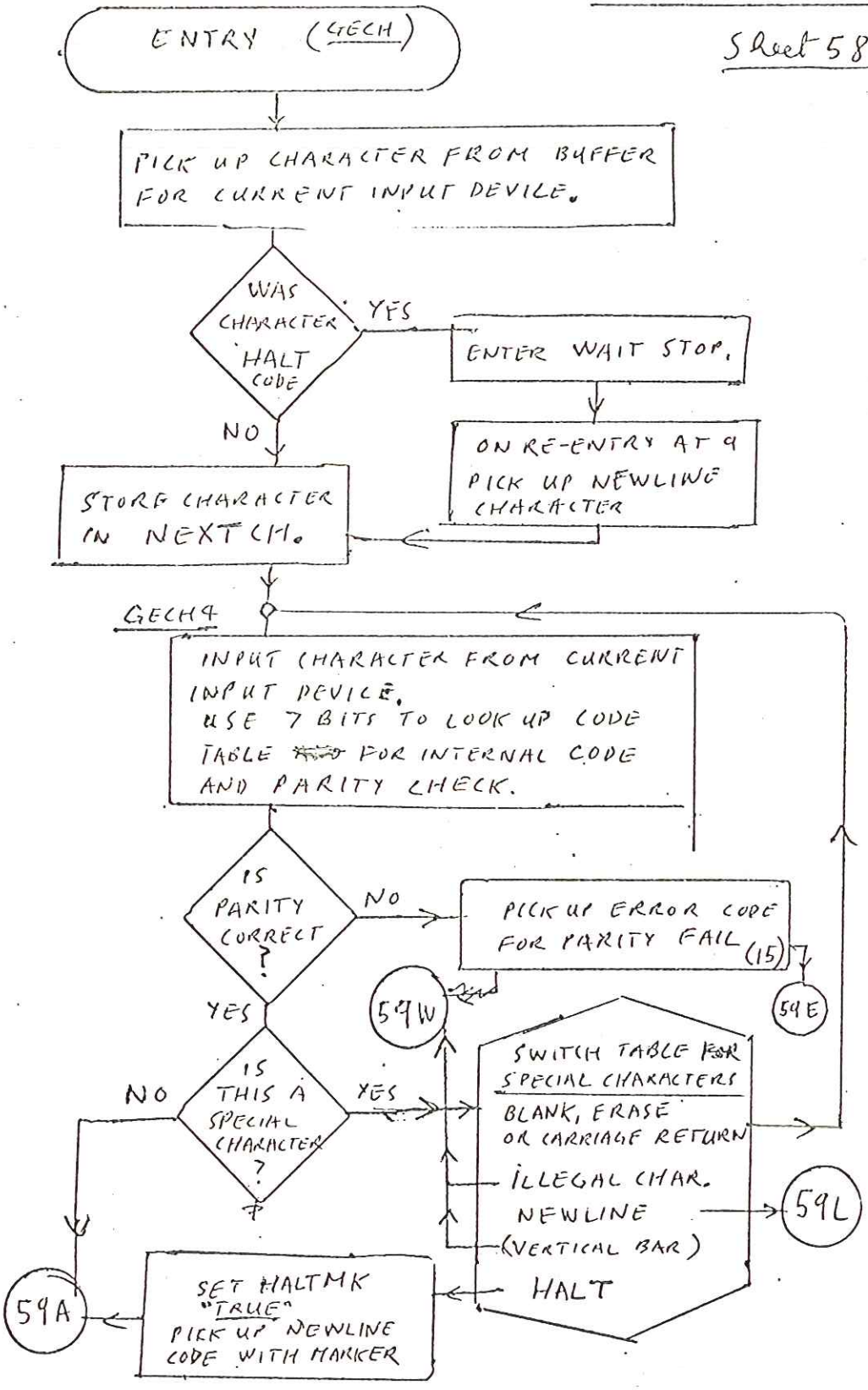


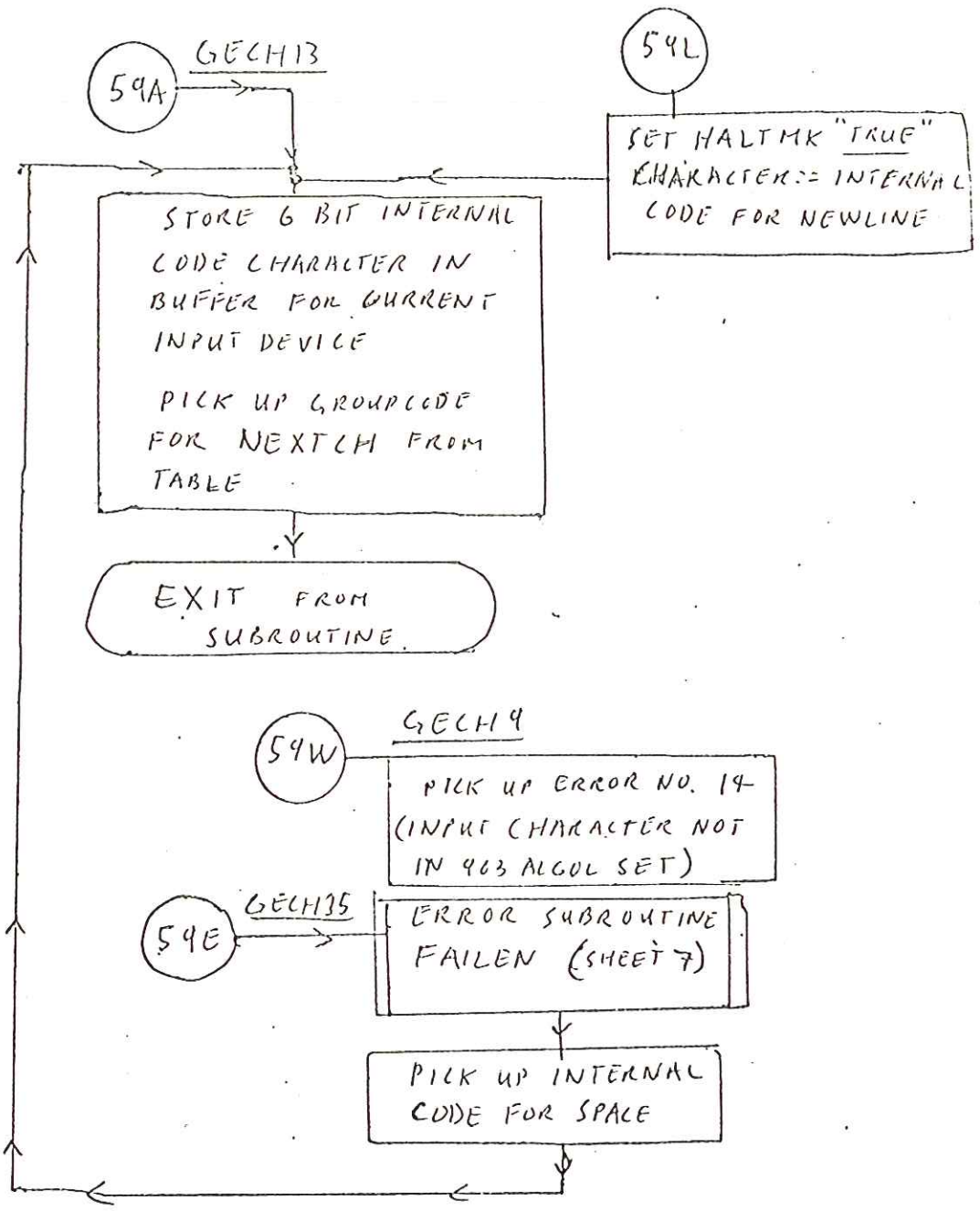

Character form is either ① Internal code ② Binary code with marker (to be output without translation)

# SUBROUTINE: GET NEXT CHARACTER

903 CODE VERSION

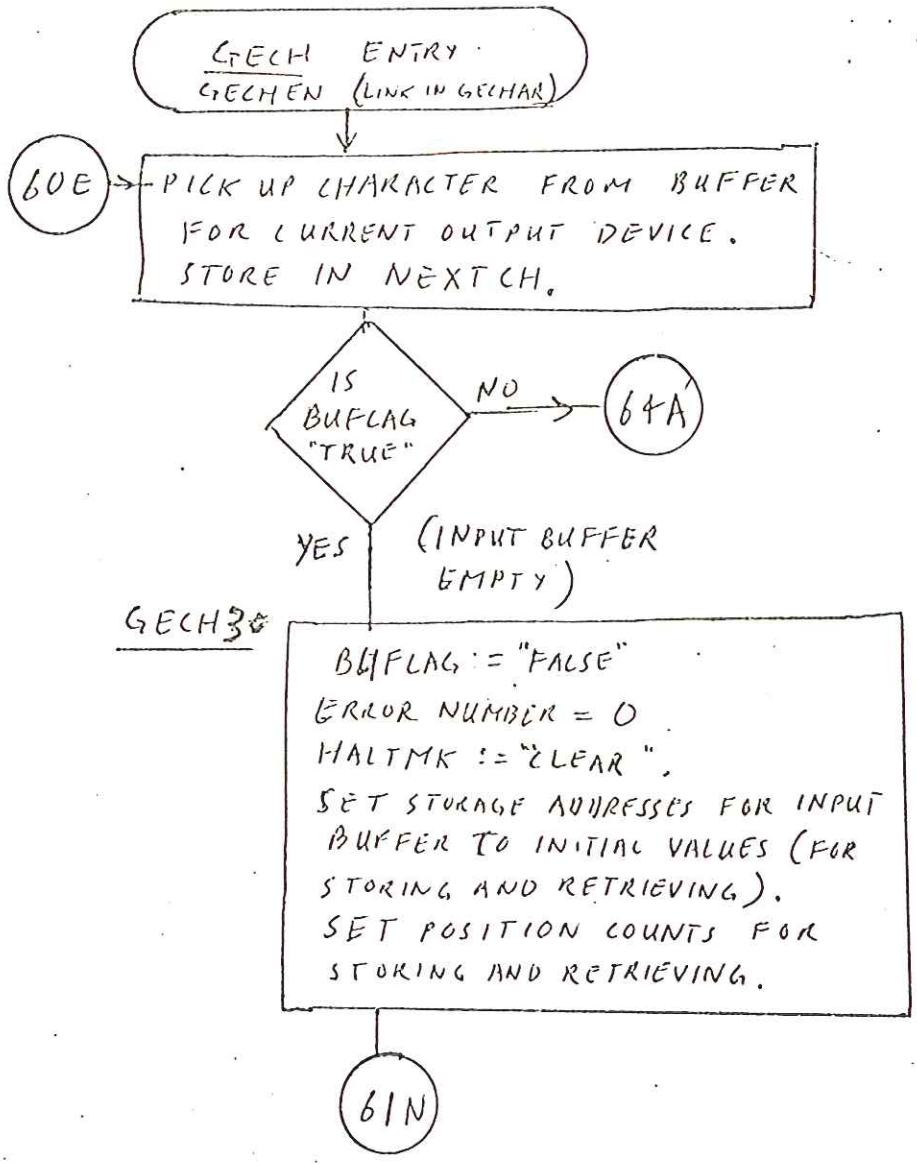


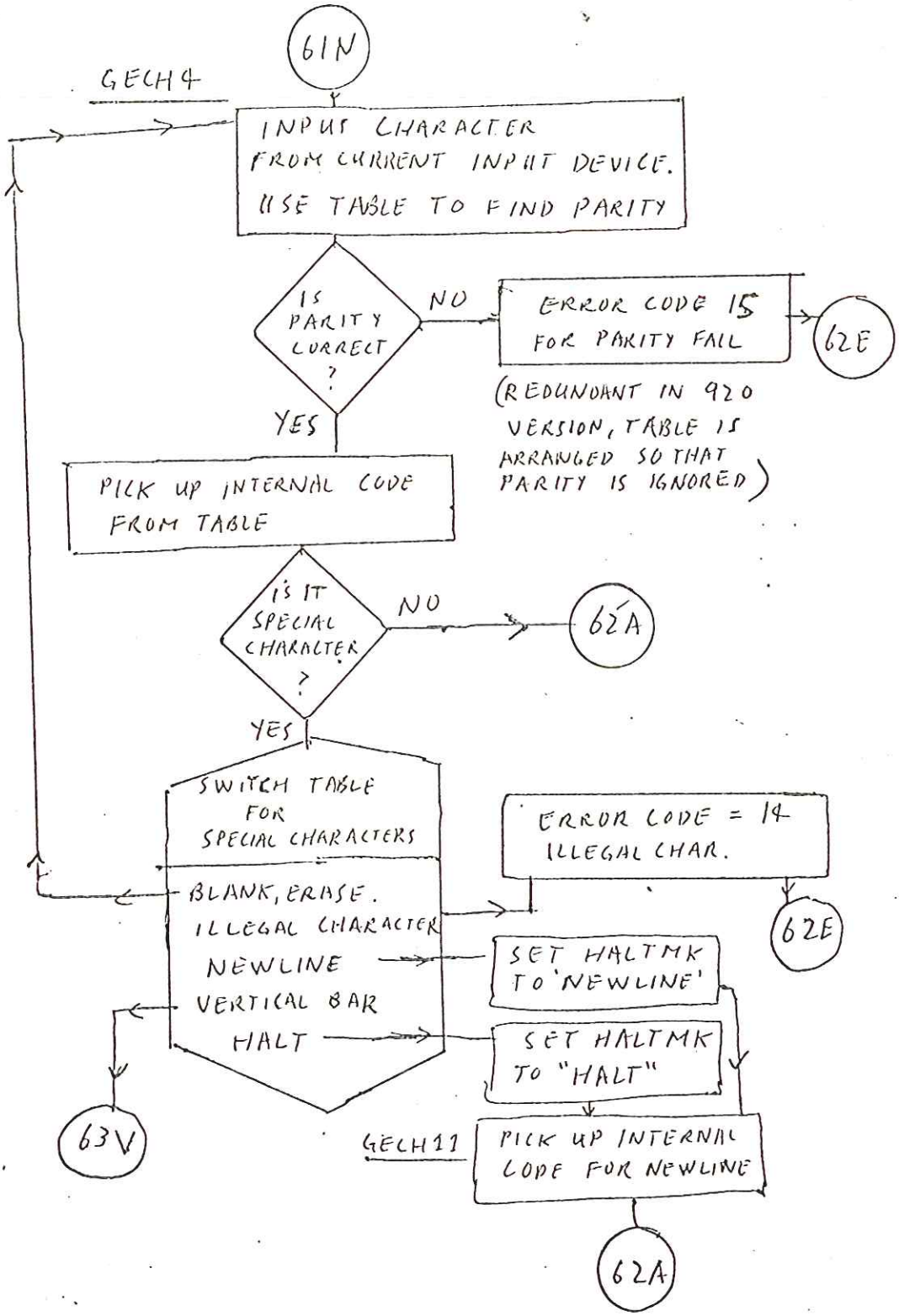


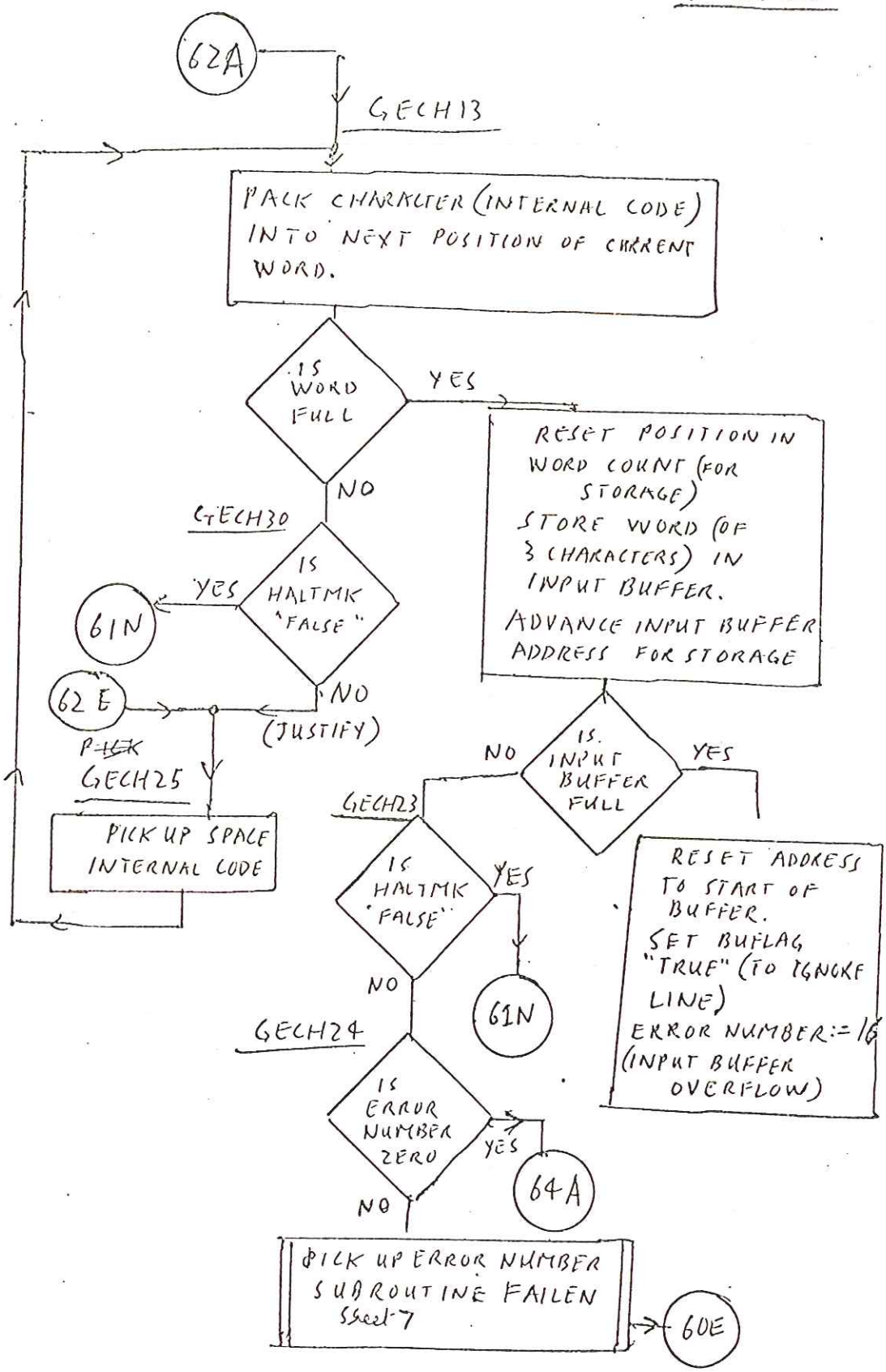

54A
GECH13
54L
SET HALTMK "TRUE"
CHARACTER := INTERNAL CODE FOR NEWLINE
STORE 6 BIT INTERNAL CODE CHARACTER IN BUFFER FOR CURRENT INPUT DEVICE
PICK UP GROUPCODE FOR NEXTCH FROM TABLE
EXIT FROM SUBROUTINE
54W
GECH9
PICK UP ERROR NO. 14 (INPUT CHARACTER NOT IN 963 ALGOL SET)
54E
GECH35
ERROR SUBROUTINE FAILEN (SHEET 7)
PICK UP INTERNAL CODE FOR SPACE

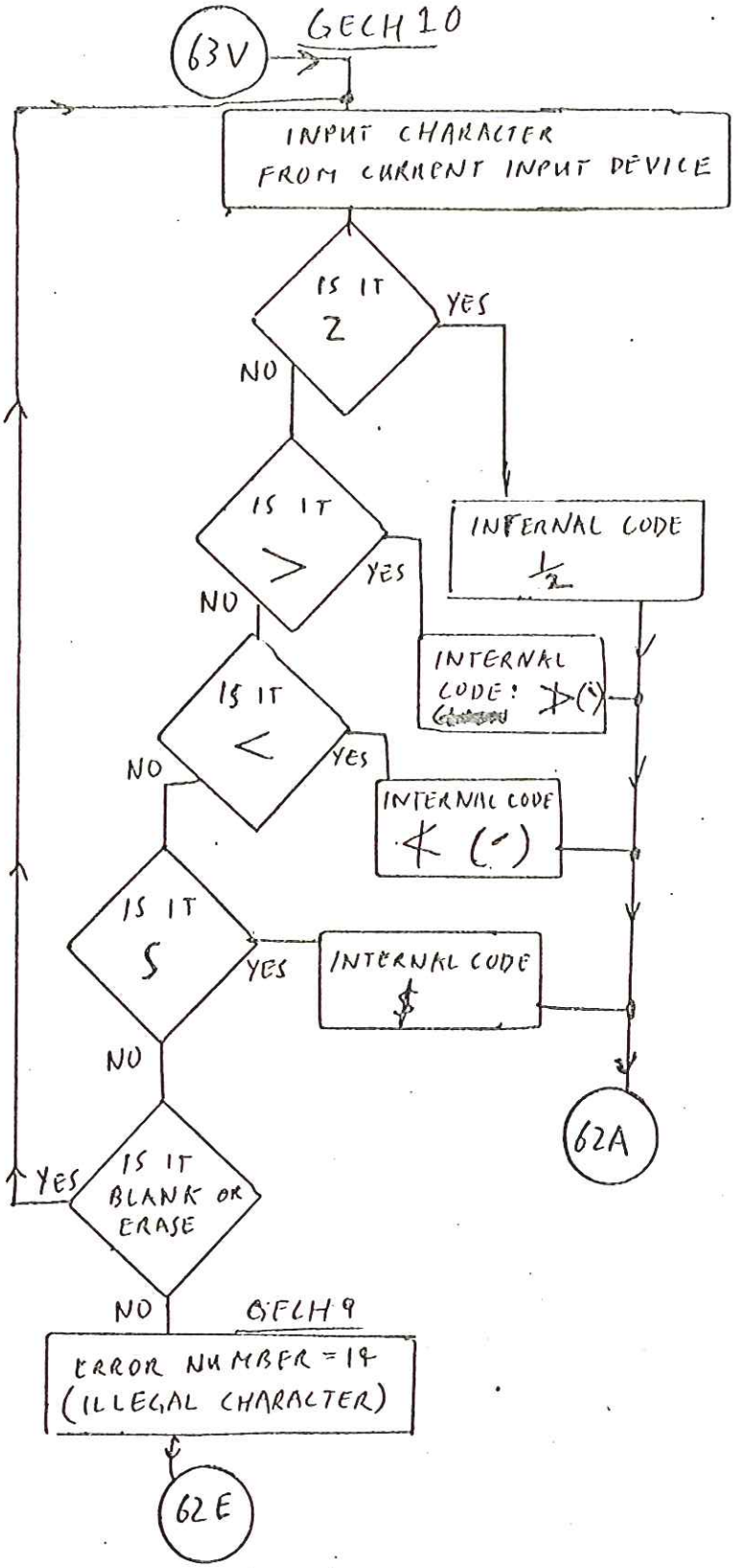
# SUBROUTINE: GET NEXT CHARACTER

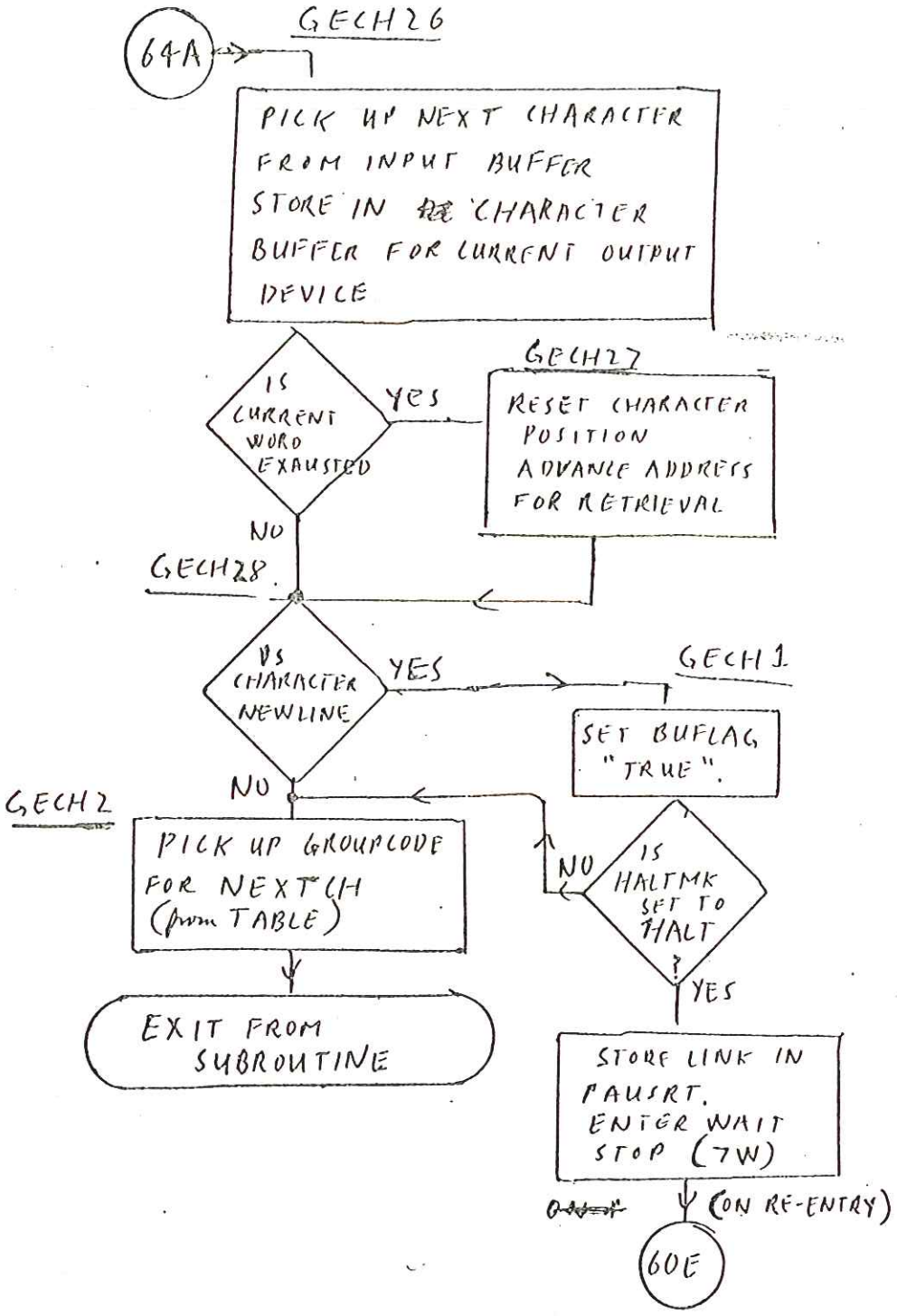
(420 CODE VERSION)




61N
GECH4
INPUT CHARACTER FROM CURRENT INPUT DEVICE. USE TABLE TO FIND PARITY
IS PARITY CORRECT ?
NO
ERROR CODE 15 FOR PARITY FAIL
62E
(REDUNDANT IN 920 VERSION, TABLE IS ARRANGED SO THAT PARITY IS IGNORED)
YES
PICK UP INTERNAL CODE FROM TABLE
IS IT SPECIAL CHARACTER ?
NO
62A
YES
SWITCH TABLE FOR SPECIAL CHARACTERS
BLANK, ERASE. ILLEGAL CHARACTER
NEWLINE
VERTICAL BAR
HALT
ERROR CODE = 14 ILLEGAL CHAR.
62E
SET HALTMK TO 'NEWLINE'
SET HALTMK TO "HALT"
63V
GECH11
PICK UP INTERNAL CODE FOR NEWLINE
62A


62A
GECH13
PACK CHARACTER (INTERNAL CODE) INTO NEXT POSITION OF CURRENT WORD.
IS WORD FULL
YES
NO
RESET POSITION IN WORD COUNT (FOR STORAGE)
STORE WORD (OF 3 CHARACTERS) IN INPUT BUFFER.
ADVANCE INPUT BUFFER ADDRESS FOR STORAGE
GECH30
IS HALTMK "FALSE"
YES
61N
62E
NO (JUSTIFY)
PACK
GECH25
PICK UP SPACE INTERNAL CODE
IS INPUT BUFFER FULL
NO
YES
GECH23
IS HALTMK "FALSE"
YES
61N
RESET ADDRESS TO START OF BUFFER.
SET BUFLAG "TRUE" (TO IGNORE LINE)
ERROR NUMBER := 16 (INPUT BUFFER OVERFLOW)
NO
GECH24
IS ERROR NUMBER ZERO
YES
64A
NO
PICK UP ERROR NUMBER SUBROUTINE FAILEN Sheet-7
60E


63V
GECH 10
INPUT CHARACTER FROM CURRENT INPUT DEVICE
IS IT Z
YES
NO
IS IT >
YES
NO
INTERNAL CODE 1/2
INTERNAL CODE: >(=)
IS IT <
YES
NO
INTERNAL CODE <(')
IS IT $
YES
INTERNAL CODE $
NO
62A
IS IT BLANK OR ERASE
YES
NO
GECH 9
ERROR NUMBER = 14 (ILLEGAL CHARACTER)
62E


64A
GECH26
PICK UP NEXT CHARACTER FROM INPUT BUFFER STORE IN CHARACTER BUFFER FOR CURRENT OUTPUT DEVICE
IS CURRENT WORD EXHAUSTED
YES
GECH27
RESET CHARACTER POSITION ADVANCE ADDRESS FOR RETRIEVAL
NO
GECH28
Is CHARACTER NEWLINE
YES
GECH1
SET BUFLAG "TRUE".
NO
GECH2
PICK UP GROUPCODE FOR NEXTCH (from TABLE)
EXIT FROM SUBROUTINE
NO
IS HALTMK SET TO HALT ?
YES
STORE LINK IN PAUSRT. ENTER WAIT STOP (7W)
(ON RE-ENTRY)
60E

# ALGOL LOADER : OVERALL FLOW