ICT 1900 FORTRAN IV

	Routine Name	Routine Called	State of Routine: Flowcharted	Coded	Punched	Name Segment	Testing	Tested	Written Specification
26	FINDCONST	LISTSCAN ITEMTYPE OUTPUT	22/1/65	10/2/65	23/2/65	FM2A			
84	ACOMPIL	CLIB OUTPUT ~~CHANGEMARK~~ ~~FORMOPERATOR~~ WORKSTORE	~~1/2/65~~ 21/2/65	~~2/2/65~~ 21/2/65	29/3/65	FR2A			
27	~~SYMBOLSTORE~~ WORKSTORE * ~~CHANGEMARK~~		~~2/2/65~~ 28/2/65	~~2/2/65~~ 28/2/65	29/3/65	FR2A			
25	CLIB	LISTSCAN ITEMTYPE OUTPUT	~~4/2/65~~ 16/3/65	~~4/2/65~~ 16/3/65	~~10/2/65~~	FE2A			
154	GETFIELD	GNC GETNUMBER FORMOPERATOR GETEXP DECBIN FDPQ	27/1/65	28/1/65	10/2/65	FP2A	31/3/65	14/4/65	
27	GETEXP *	GNC GETNUMBER	28/1/65	28/1/65	10/2/65	FP2A		14/4/65	
50	FORMOPERATOR *	GNC	28/1/65	28/1/65	10/2/65	FP2A	31/3/65	14/4/65	
35	GETNUMBER	GNC	28/1/65	28/1/65	10/2/65	FP2A	31/3/65	14/4/65	
	DECBIN	—	old compiler	—	—	FP2A	31/3/65	~~14/4/65~~	
23	GNC *	PRELIMINARY LINKS	28/1/65	28/1/65	10/2/65	FP2A	31/3/65	14/4/65	
	FDPQ *					FP2A			
89	FOPTR	~~SYMBOLSTORE~~	3/2/65 corrected 15/2/65	17/2/65	23/2/65	FM2A			
39	MNLI	SETM LISTSCAN ~~GENSPACE~~ CIF	~~3/2/65~~ 15/2/65	16/2/65	23/2/65	FM2A			
10	GENSPACE *	—	3/2/65	16/2/65	23/2/65	~~FM2A~~ FN2A			
12	SETM	—	3/2/65	16/2/65	23/2/65	FM2A			
7	ITEM TYPE	—	old compiler	~~15/2/65~~	~~23/2/65~~	FM2A	—	—	
45	MALI	OUTPUT SETM	12/2/65	~~12/2/65~~	23/2/65	FM2A			
127	UDML	NEGATE GETFIELD FINDCONST MNLI	15/2/65	16/2/65	23/2/65	FM2A			
32	* CIF	—	15/2/65	16/2/65	23/2/65	FM2A			
98	/INTFC *	/INTFC —	—	16/2/65	23/2/65	/INTPC			
28	FORMPN (general flow)	GETFIELD UDML FOPTR COMPIL	15/2/65	16/2/65	23/2/65	FM2A			

* - Routine not in old compiler

ICT 1900 FORTRAN IV

	Routine Name	Routine Called	State of Routine: Flowcharted	Coded	Punched	Tape Name Segment	Testing	Tested	Written Specification
37	MMLI	LISTSCAN ITEMTYPE SETM MALI MNLI	old compiler (18/2/65)	18/2/65	23/2/65	FM2A			
6	NEGATE *	——	——	19/2/65	23/2/65	FM2A			
31	LOADACC *	ACØMPIL MVLI	21/2/65	25/2/65	12/3/65	FN2A			
11	STØRACC *	ACØMPIL	21/2/65	25/2/65	12/3/65	FN2A			
37	LOADX3 *	ACØMPIL MVLI	22/2/65	25/2/65	12/3/65	FN2A			
18	STØRX3 *	ACØMPIL	21/2/65	25/2/65	12/3/65	FN2A			
93	BINARITH (BCØMPIL)	ACØMPIL ~~LØADACC~~ ~~STØRACC~~ ~~LØADX3~~ ~~STØRX3~~ BINSTANDARD	23/2/65	25/2/65	12/3/65	FN2A			
8	UMINUS	UNSTANDARD ~~LOADACC~~ ~~STORACC~~ ACØMPIL	28/2/65 ~~25/2/65~~	28/2/65 ~~25/2/65~~	12/3/65	FN2A			
80	FUNCTN	STØRACC STØRX3 ANALYZE ~~[illegible]~~ FUNSEARCH ØUTPUT ACØMPIL	25/2/65	~~25/2/65~~ 26/2/65	12/3/65	FN2A			
58	ANALYZE	FUNSEARCH ~~LISTSCAN~~ ~~ITEMTYPE~~ FINDCONST ØUTPUT	26/2/65	27/2/65	12/3/65	FN2A			
41	RELBIN *	BINARITH ACØMPIL ~~LISTSCAN~~ ~~ITEMTYPE~~ ØUTPUT FINDCONST	27/2/65	27/2/65	12/3/65	FN2A			
26	BINSTANDARD *	LØADACC STØRACC LØADX3 STØRX3	27/2/65	27/2/65	12/3/65	FN2A			
32	LØGBIN *	BINSTANDARD ACØMPIL	28/2/65	28/2/65	12/3/65	FN2A			
13	UNSTANDARD *	LØADACC STØRACC	28/2/65	28/2/65	12/3/65	FN2A			
10	LØGNØT *	UNSTANDARD ACØMPIL	28/2/65	28/2/65	12/3/65	FN2A			

* - routine not in old compiler

ICT 1900 FORTRAN IV

	Routine Name	Routine Called	State of Routine: Flowcharted	Coded	Punched	Tape Name Segment	Testing	Tested	Written Specification
9	CØMPIL	BINARITH UMINUS FUNCTN EQUALS RELBIN LØGBIN LØGNOT IFQ	28/2/65	1/3/65	11/3/65	FN2A			
50	EQUALS	LØADACC STØRACC STØRX3 ACØMPIL MVLI	1/3/65	1/3/65	12/3/65	FN2A			
23	MVLI *	ITEMTYPE GENSPACE	8/3/65	8/3/65	29/3/65	FN2A			
60	ASF *	GETFIELD LISTSCAN LISTDUMMY PRØCØMPIL FØRMPN ACØMPIL	9/3/65	13/3/65	27/3/65	FN2A			
	PRELIMINARY LINESCAN	INPUT	10/3/65	✓	✓	FP2A		14/4/65	11/3/65
	STATEMENT *	STS ARITHMETIC ~~[illegible]~~ Routines which compile all statements except END & FINISH	11/3/65	16/3/65	22/3/65	FE2A			
13	ARITHMETIC (LIF)	FØRMPN	11/3/65	15/3/65	18/3/65	FH2A			
68	IFQ *	GETFIELD LISTSCAN ITEMTYPE MALC ASF LIF AIF	11/3/65	12/3/65	27/3/65	FN2A			
21	LIF *	~~[illegible]~~ ~~[illegible]~~ ØUTPUT STATEMENT UNSTANDARD	11/3/65	12/3/65	27/3/65	FN2A			
	~~PROCESS STATEMENT~~ PRØCESS SEGMENT (PRØCESS STATEMENT)	INITIALIZE PRELIMINARY LINESCAN DE2 LISTSCAN ITEMTYPE STATEMENT ØUTPUT ENDE	11/3/65	16/3/65	22/3/65	FE2A			
67	AIF *	UNSTANDARD ACØMPIL GETFIELD IFGEN	12/3/65	12/3/65	27/3/65	FN2A			
23	GETLABEL *	GNC GETNUMBER	14/3/65	14/3/65	17/3/65	FP2A			14/3/65

* - routine not in old compiler

ICT 1900 FORTRAN IV

	Routine Name	Routine Called	State of Routine: Flowcharted	Coded	Punched	~~Tape~~ Name Segment	Testing	Tested	Written Specification
48	TYPE	GETFIELD LISTSCAN DIMENS	14/3/65	14/3/65	18/3/65	FK2A			
36	ASSIGN*	GETLABEL LABELLIST ØUTPUT GETFIELD FINDINTVAR ACØMPIL	14/3/65	15/3/65	18/3/65	FH2A			
44	FUNSUB	GETFIELD LISTSCAN OUTPUT MULI LISTDUMMY PRØCØMPIL INITIALIZE	14/3/65	14/3/65	25/3/65	FF2A			
33	RETURN	ACØMPIL ØUTPUT	14/3/65	13/3/65	25/3/65	FF2A			
8	DIMENSION	GETFIELD DIMENS LISTSCAN	16/4/65	16/4/65	—	FK2A	—	—	~~Old Compiler~~
68	DIMENS	—	—	16/4/65	—	FK2A	—	—	Old Compiler
51	QIDENT*	LISTSCAN SETM ITEMTYPE GENSPACE DIMENS MULI OUTPUT	16/4/65	~~15/4/65~~ 16/4/65		FK2A	—	—	~~Old Comp~~
57	CØMMØN	GETFIELD LISTSCAN ITEMTYPE OUTPUT QIDENT	16/4/65	16/4/65	—	FK2A	—	—	~~Old Compiler~~
31	~~PUBLIC~~	—	—	—	—	FK2A	—	—	~~Old Compiler~~
58	GO	GETFIELD FINDINTVAR LABELLIST OUTPUT ACOMPIL LABELS	15/3/65	15/3/65	18/3/65	FH2A			
42	CALL	GETFIELD LISTSCAN ITEMTYPE ØUTPUT FØRMØN ACØMPIL	15/3/65	15/3/65	18/3/65	FH2A			
	STS	—	17/3/65	17/3/65	30/3/65	FE2A			
30	MASTER	GETFIELD ØUTPUT LISTSCAN CLIB	18/3/65	18/3/65	25/3/65	FF2A			
	PRØCØMPIL	—	—	—	—	—	—	—	—
	EXTERNAL	—	—	—	—	—	—	—	—

* routine not in old compiler

ICT 1900 FORTRAN IV

ROUTINE NAME	ROUTINE CALLED	STATE OF ROUTINE						
		Flowcharted	Coded	Punched	Tape Name Segment	Testing	Tested	Written Specification
IØPER	GETFIELD FINDINTVAR ACØMPIL	1/4/65	3/4/65	6/4/65	FJ2A			
FINDINTVAR	LISTSCAN ITEMTYPE GENSPACE	1/4/65	3/4/65	7/4/65	FH2A			
IØRW	CLIB IØPER GETFIELD LISTSCAN ITEMTYPE MHLI ACØMPIL	1/4/65	3/4/65	6/4/65	FJ2A			
IØLIST	GETFIELD LISTSCAN ACØMPIL ØUTPUT DØ1 DØ2 MHLI FØRMPN	4/4/65	4/4/65	6/4/65	FJ2A			
FØRMAT	ØUTPUT	4/4/65	4/4/65	6/4/65	FJ2A			
IØMISC *	ACØMPIL IØPER	6/4/65	5/4/65	6/4/65	FJ2A			
LABELS	old compiler		5/4/65	7/4/65	FH2A			
DØ2	ACØMPIL LISTSCAN	6/4/65	7/4/65		FI2A			
DS	GETFIELD FINDINTVAR FINDCONST FORMPN UNSTANDARD OUTPUT	6/4/65	7/4/65		FI2A			
DØ1	LISTSCAN GETLABEL GETFIELD FINDINTVAR FORMPN DS	6/4/65	7/4/65		FI2A			
DØ	DØ1	7/4/65	7/4/65		FI2A			
BLOCK DATA	—	14/4/65	14/4/65		FF2A			
ERROR	LØADBUF INITERR ~~ERROR~~ LIST P.L.S.	26/4/65						
PAUSE } STOP }	GETFIELD ACØMPIL	27/4/65						

* not in old compiler

ICT 1900 FORTRAN IV

Routine Name	Routine Called	State of Routine						
		Flowcharted	Coded	Punched	Tape Name	Testing	Tested	Written Specification
EQUIVALENCE	GETFIELD LISTSCAN RELARAY ITEMTYPE SETM MALI MVLI CLIB OUTPUT GENSPACE	5/5/65	7/5/65	9/5/65				
DATA	GETFIELD LISTSCAN ITEMTYPE MALI SETM GENSPACE RELARAY NEGATE CLIB OUTPUT	11/5/65	12/5/65	14/5/65				
RELARAY	ITEMTYPE SETM GENSPACE GETFIELD	11/5/65	12/5/65	14/5/65				

22/1/65

FORTRAN IV

FM

FINDCONST

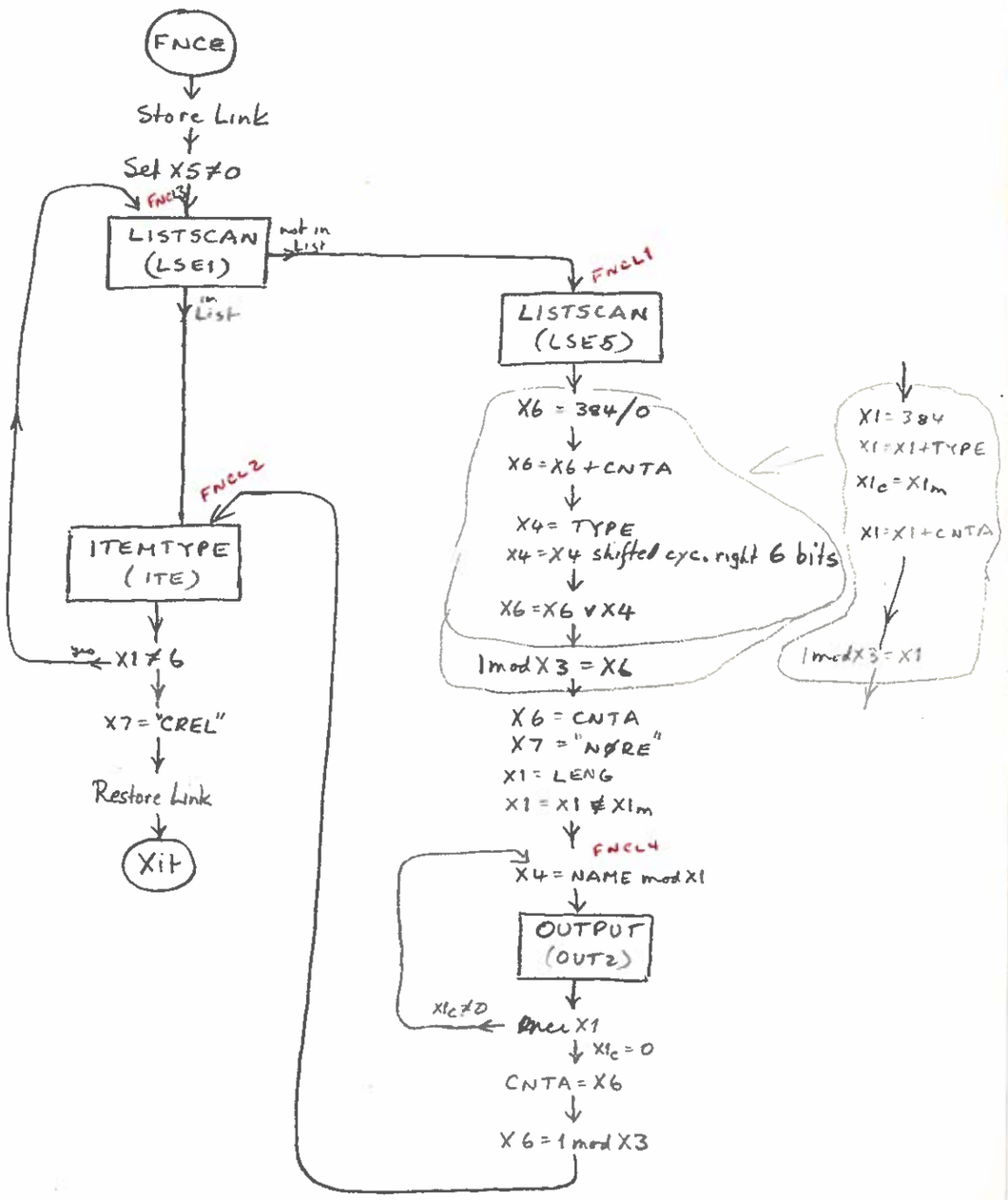
TYPE = 8 – Fxpt.
= 16 – Float Pt
= 24 – D. P
= 32 – Complex
= 40 – Logical

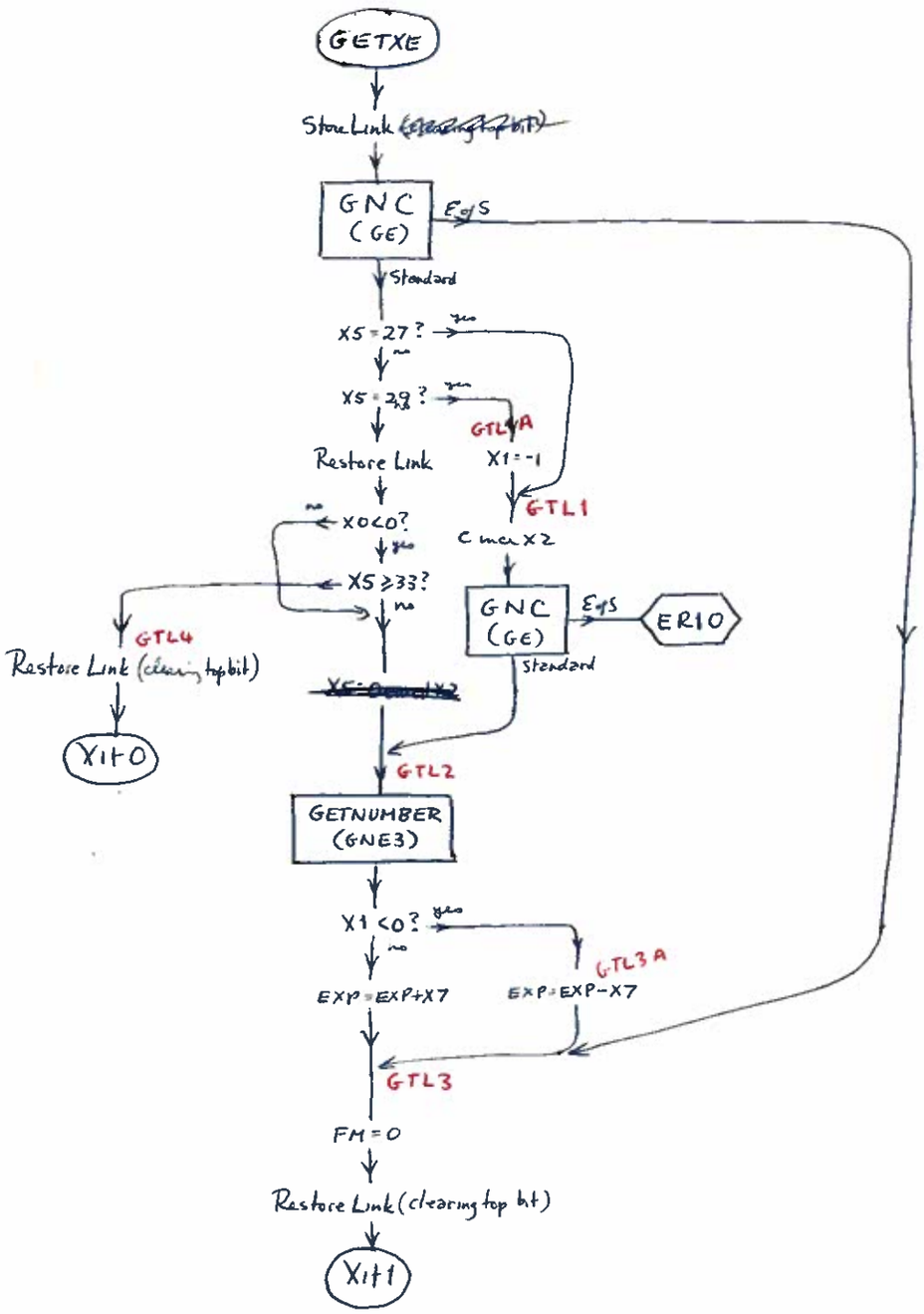
LENG = C/? C = length of constant

NAME etc – name of constant

Note TYPE to be generated by FORMPN etc

Routine generates main List item for constant & outputs constant into "constants" space.





27

Entry Pt. GETXE Routine will exit with OVR clear even if set before entry

Used in GETFIELD to find the exponent after E or D. Exponent as integer in EXP on exit, X2 pointing to character after terminator. Exit if exponent found to 2nd word after Cue. FM=0 if exponent found.

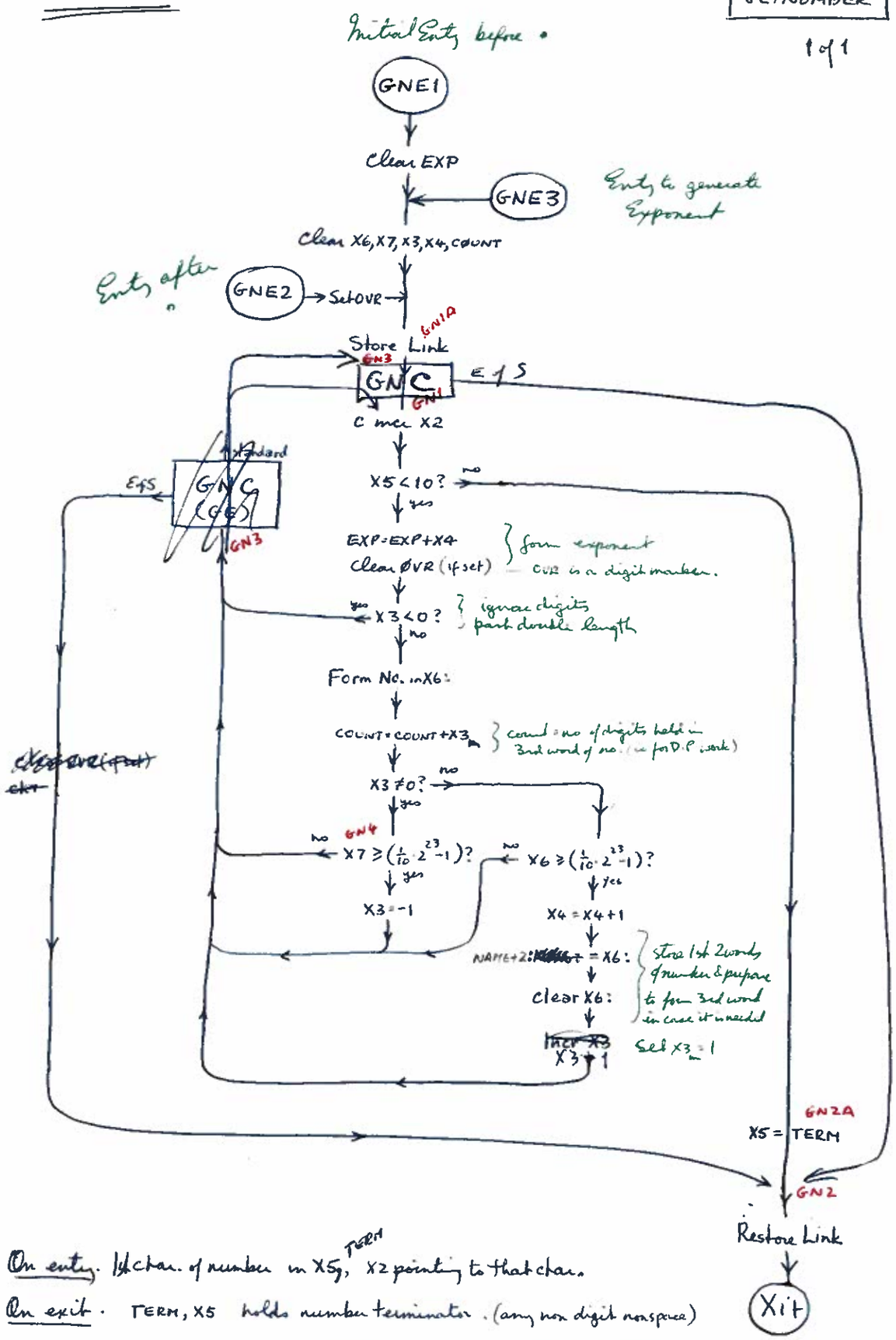
If not an exponent but .E or .D, exit at 1st word after Cue with X2 pointing to 2nd character after "."

27/1/65

FORTRAN IV

GETNUMBER

1 of 1



On entry. If char. of number in X5, TERM, X2 pointing to that char.

On exit. TERM, X5 holds number terminator. (any non digit nonspace)

X3 = 0 X6: holds number, EXP holds decimal exponent, COUNT = 0

X3 ≠ 0 NWS: holds number X7 holds remainder of number.
EXP holds decimal exponent COUNT holds no. of decimal digits in X7

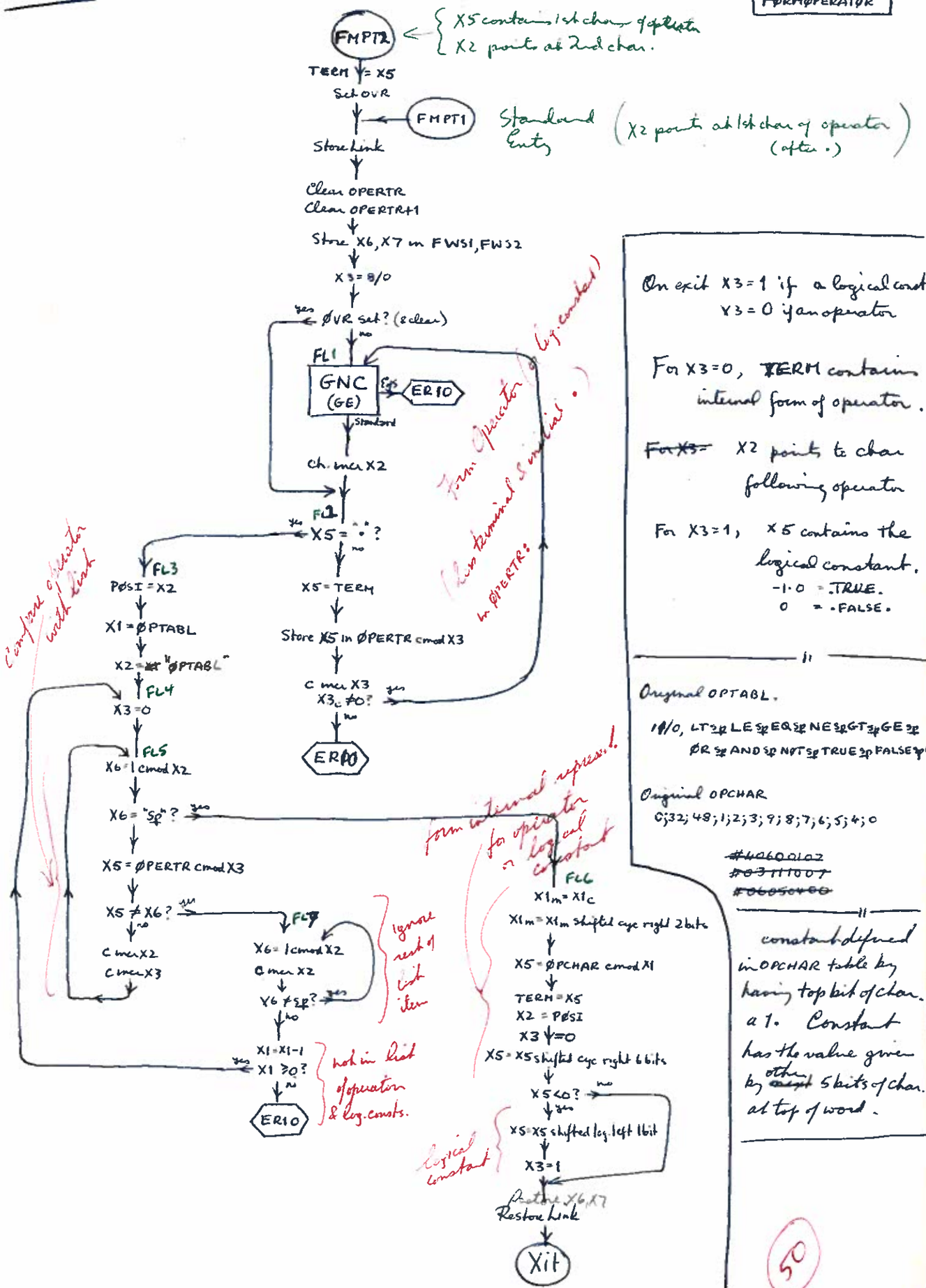
36

If OVR set, terminator immediately follows.

27/11/00

FORTRAN IV

FØRMØPERATØR



On exit X3 = 1 if a logical const.
X3 = 0 if an operator

For X3 = 0, TERM contains internal form of operator.

~~For X3 =~~ X2 points to char following operator

For X3 = 1, X5 contains the logical constant.
-1.0 = .TRUE.
0 = .FALSE.

Original OPTABL.

1Ø/0, LT sp LE sp EQ sp NE sp GT sp GE sp
ØR sp AND sp NOT sp TRUE sp FALSE sp Ø

Original OPCHAR

C; 32; 48; 1; 2; 3; 9; 8; 7; 6; 5; 4; 0

~~#U460 0102~~
~~#03111007~~
~~#0605 0700~~

constant defined in OPCHAR table by having top bit of char. a 1. Constant has the value given by ~~eight~~ 6 bits of char. at top of word.

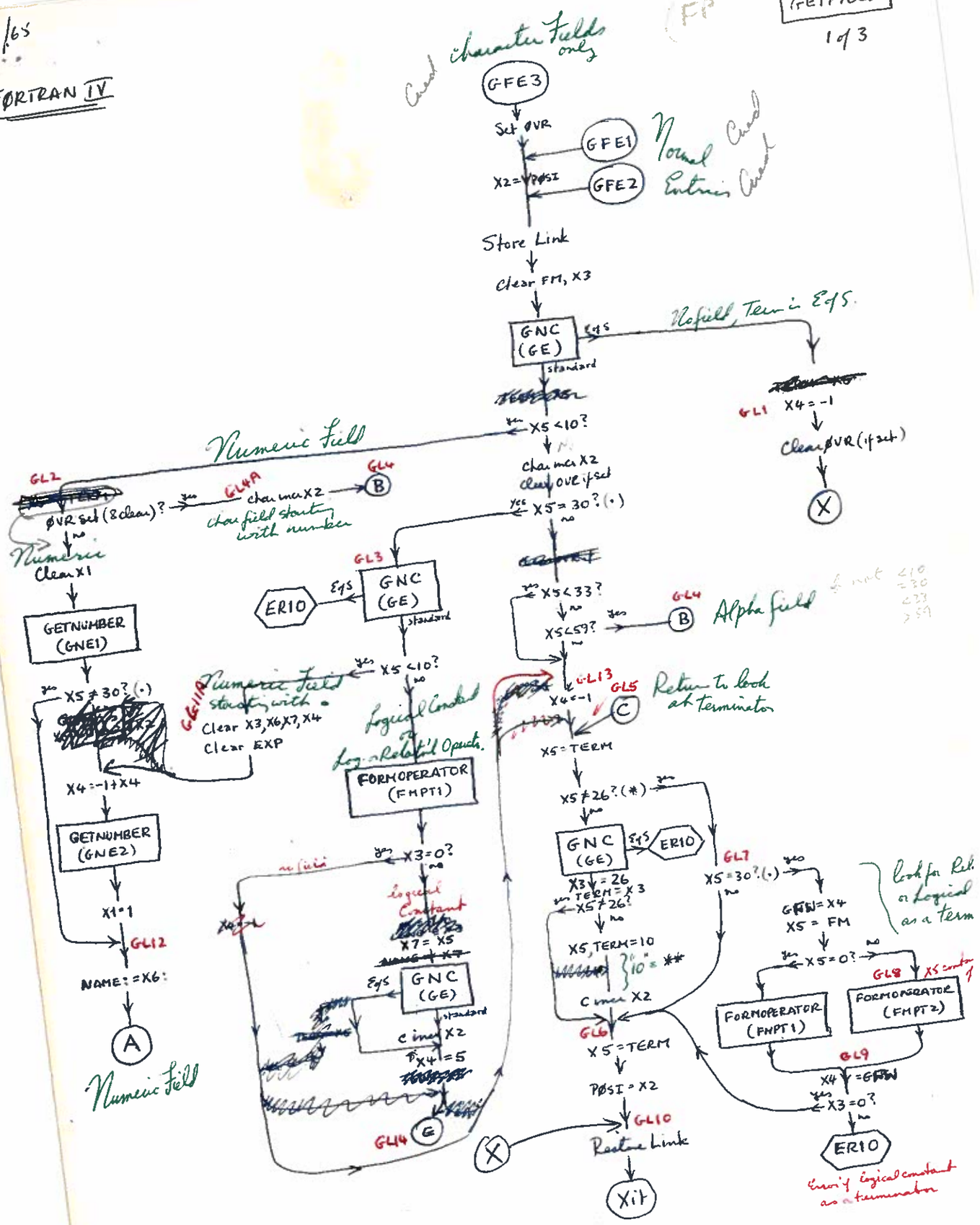
50

27 1/65

FORTRAN IV

GETFIELD

1 of 3



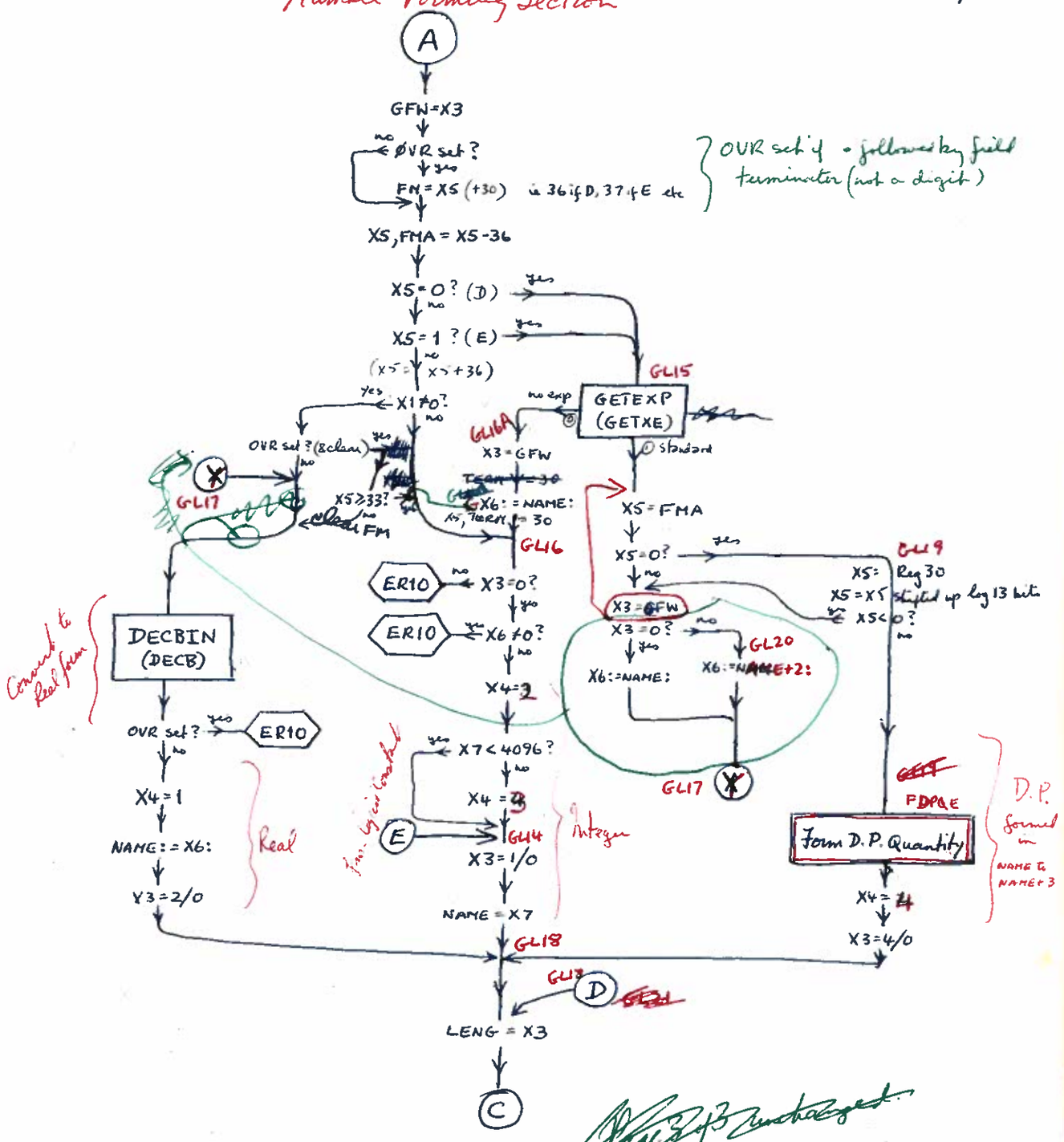
X4 = -1 no field
X4 = 0 Alpha
X4 = 1 fl pt
X4 = 4 Double precision
X4 = 2 Small Integer
X4 = 3 Integer
X4 = 5 Logical constant

(.TRUE.), (.FALSE.) (.TRUE. = -1.0, .FALSE. = 0)

26/1/65 FORTRAN IV

GETFIELD
2 of 3

Number Forming Section



Page 3 of 3 Continued

At (C), constant is in NAME, X7 for Integer — X4 = ~~3~~ 2 if no. < 4096; X4 = ~~4~~ 3

NAME, NAME+1 }, X6, X7 } for REAL — X4 = 1

NAME → NAME+3 for D.P — X4 = ~~4~~ 4?

Length of Constant in counter of LENG. $LENG_m = 0$

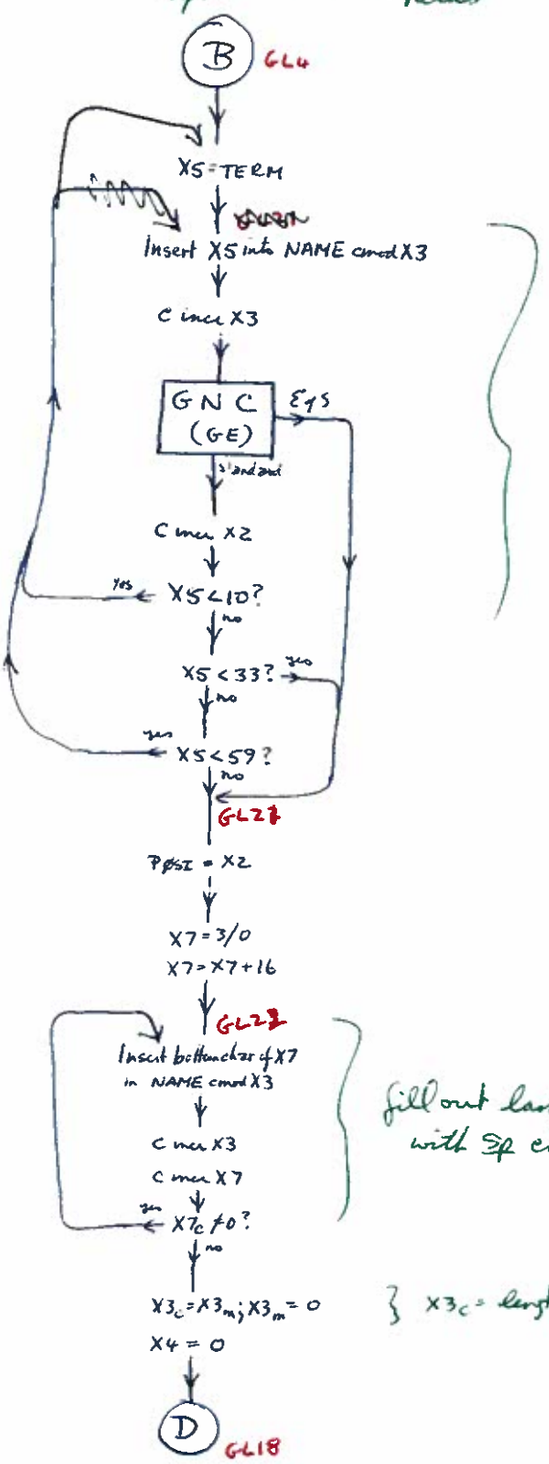
FM = 0 for standard terminator i.e. not . E or . D or . letter.

FM = letter if terminator is . letter.

For FM=0, X2 points to character following the terminator

FM≠0 X2 points to character following the letter.

Alpha & Character Fields



149

1/2/65

ACOMPIL 1 of 2

ACOM — Standard Entry. - uses QUERY table

Store Link

X2 = X2 + "QUERY"

XTINST, X6 = 1 mod X2 — start of correct TINST section

X2 = 0 mod X2

X7 = X2

X7m = X7c

XTREL, X7 = X7 + "TREL" — start of correct TREL section

X2 = X2 + X3

XCMTA, X2 = 0 mod X2 — correct CMTA entry

X2 = 0? yes → ERIO — Illegal operation - operation not defined

no

X1 = PNP1

X2 = X2 shifted log up 1 bit — X2 < 0 if PN to be changed

X2 ≥ 0? yes

no

X2 = X2 shifted log up 1 bit — X2 < 0 if previous PN to be changed

X2 ≥ 0? yes

no

X1 = X1 + 1

PNP1 = X1

Store X2 / X2 = 0 mod X1 / WKSEG / restore X2 — check whether CMTA has to be documented.

X3 = X2

X3 = X3 shifted cyc left 2 bits

X3 = X3 & #60000001

X3 = PNLIST or X3 — X3 = skeleton PN list item for result.

X3 = X3 shifted cyc right 6 bits

X2 = X2 shifted log up 1 bit

X2 = X2 & #00700000

X2 = X3 ∪ X2 — X2 = PN list item including mode

WORKSTORE (WKSEG) — on exit, X2 = complete PN list item (if WS used, routine assigns correct WS)

X1 = PNP1

Store X2 in 0 mod X1 — store PN item

Accum, X3, MARK amended according to PN list item generated.

ACML1

A

Amend MARK, MARK+1 if reqd.

X1 = X1 shifted cyc Right 3 bits

X1 < 0?

Clear MARK mod X1

X7 = PNP1

Shift X1 cyc Left 3 bit

X1 = X1 & 3

X1 = 0? yes

Store X7 in MARK-1 mod X1

X1 = X1 - 1

X1 < 0?

Clear MARK mod X1

X4 = 1?

Clear MARK

Clear MARK+1

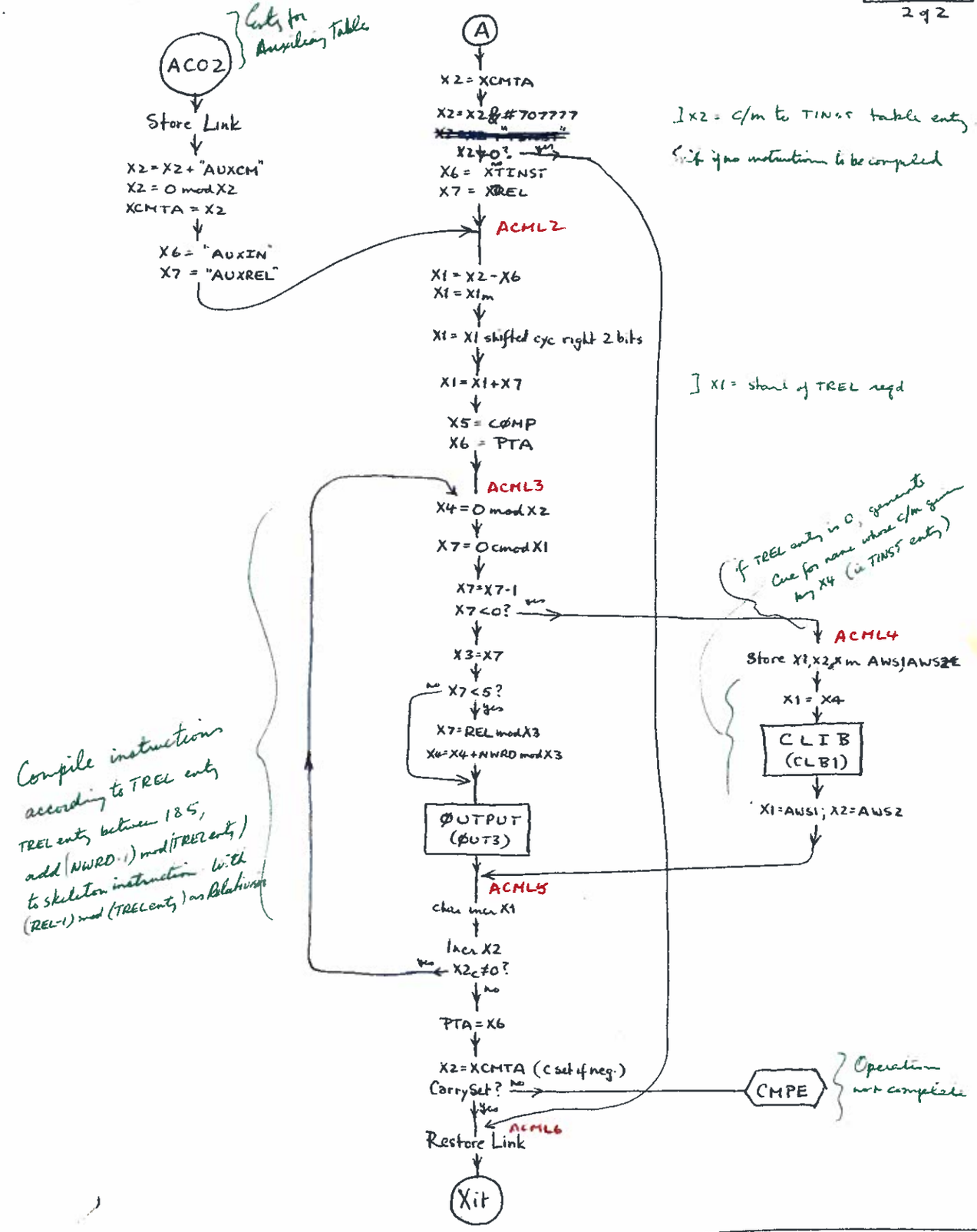
PNLIST 32,32,19,18;11,11,13,13

MLIST #c5100602;#10000610

MARK = accum mark

MARKM = X3 mark

73



PNLIST table. - character table which holds characteristics of PN items. Characters are in the order

1st word
- 0 - Accum
- 1 - X3 = addr of result
- ~~2 - X2 = addr of result~~
- ~~3 - Float. Hardware Accum.~~

2nd word
- ~~4 - X3 & WS = addr of result~~
- ~~5 - X2 & WS = addr of result~~
- 6 - WS = result
- 7 - WS = address of result

32	32
18	18
~~19~~	19
~~20~~	20
~~16~~	16
~~17~~	17
11	11
13	13

32	32
19	19
18	18
11	11
11	11
13	13
13	13

0 1	Accum
1 2	X3 = addr of result if Arrays op.
1 3	
24	WS = result
25	
36	WS = Addr of quantity (if Arrays op store)
37	

C+0

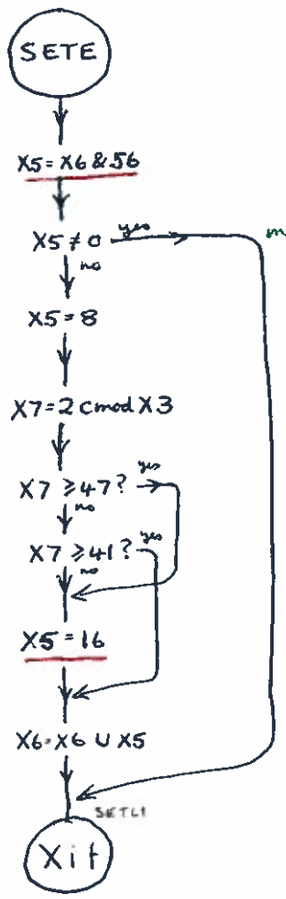
FORTRAN IV

3/2/65

FM

SETM

1 of 1



} X7 = 1st letter of name. (typ 2 bit of X3=0)

Set Integer or Real depending on 1st letter of name

X6 = updated attributes.

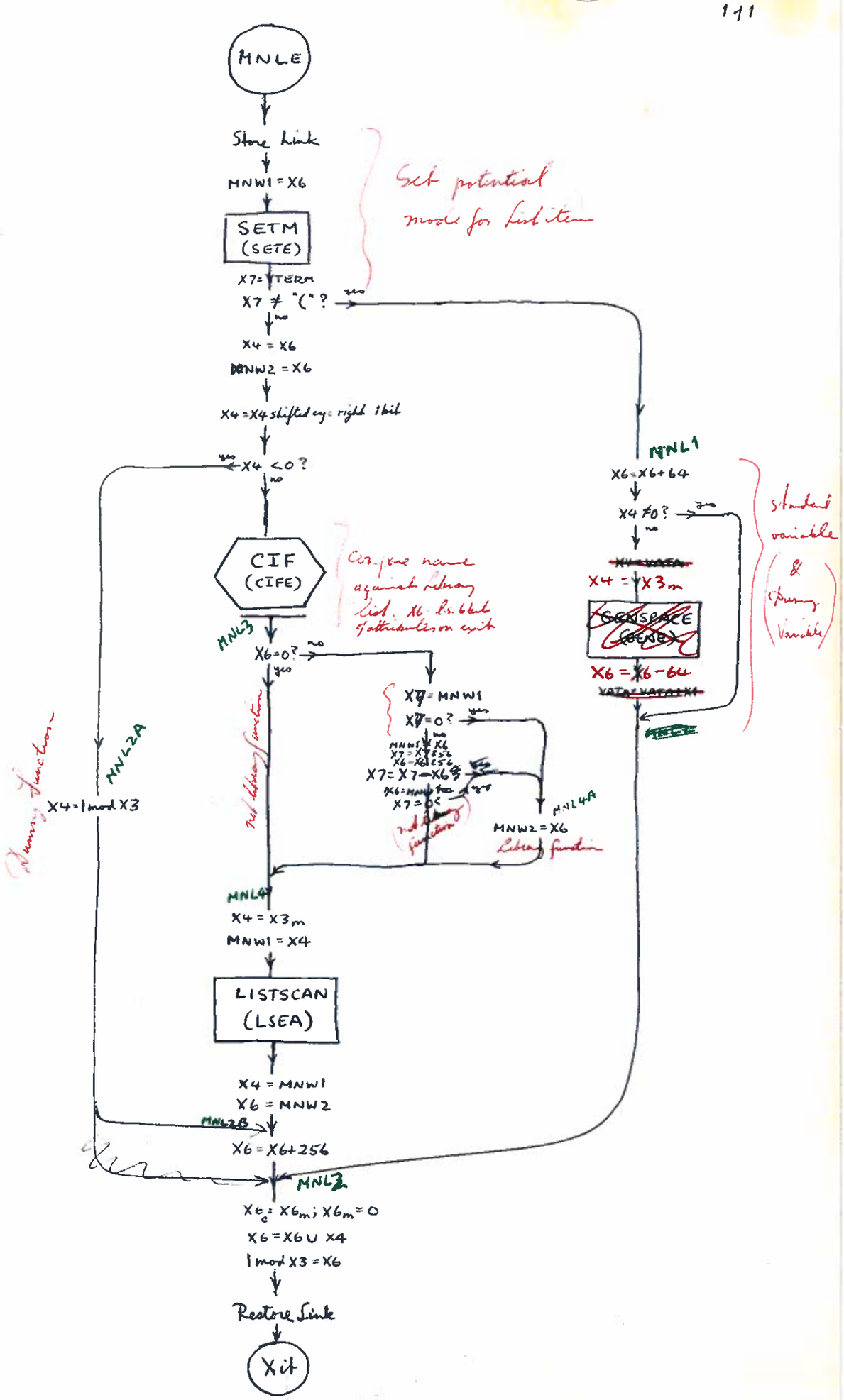
15/2/65

FØRTRAN IV

FM

MNLI

111

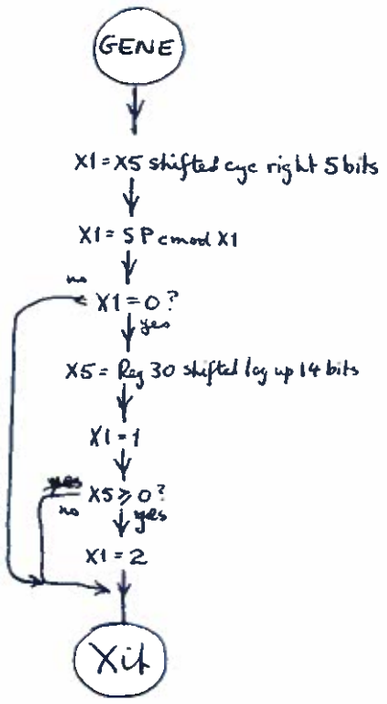


3/2/65

FN

GENSPACE

1q1



extract size of space reqd to X1
If X1=0, size under switch control (switch 14)

Switch 14 on gives 1 words
Switch 14 off gives 2 words

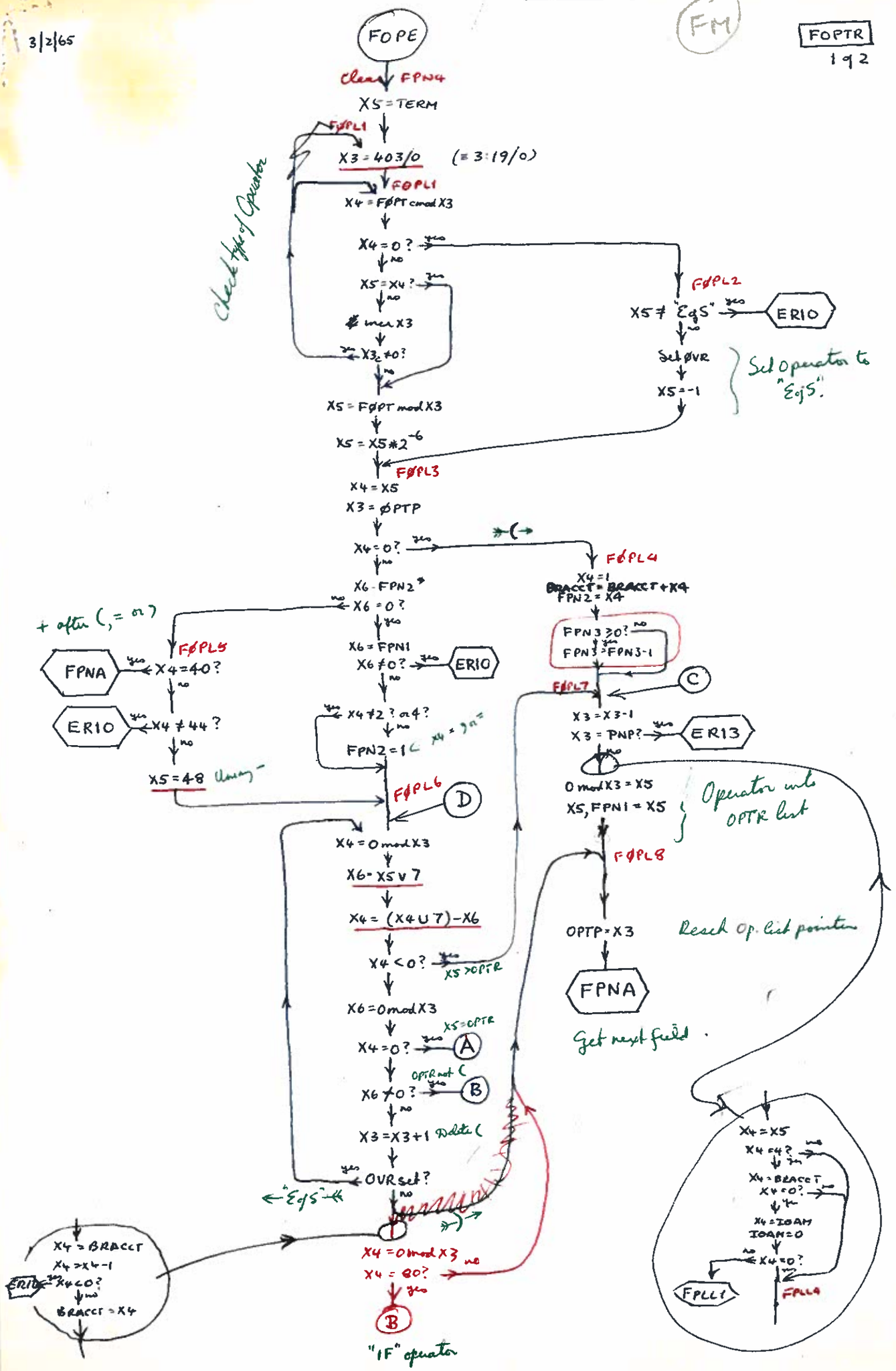
GENSPACE requires a table (2 words.)

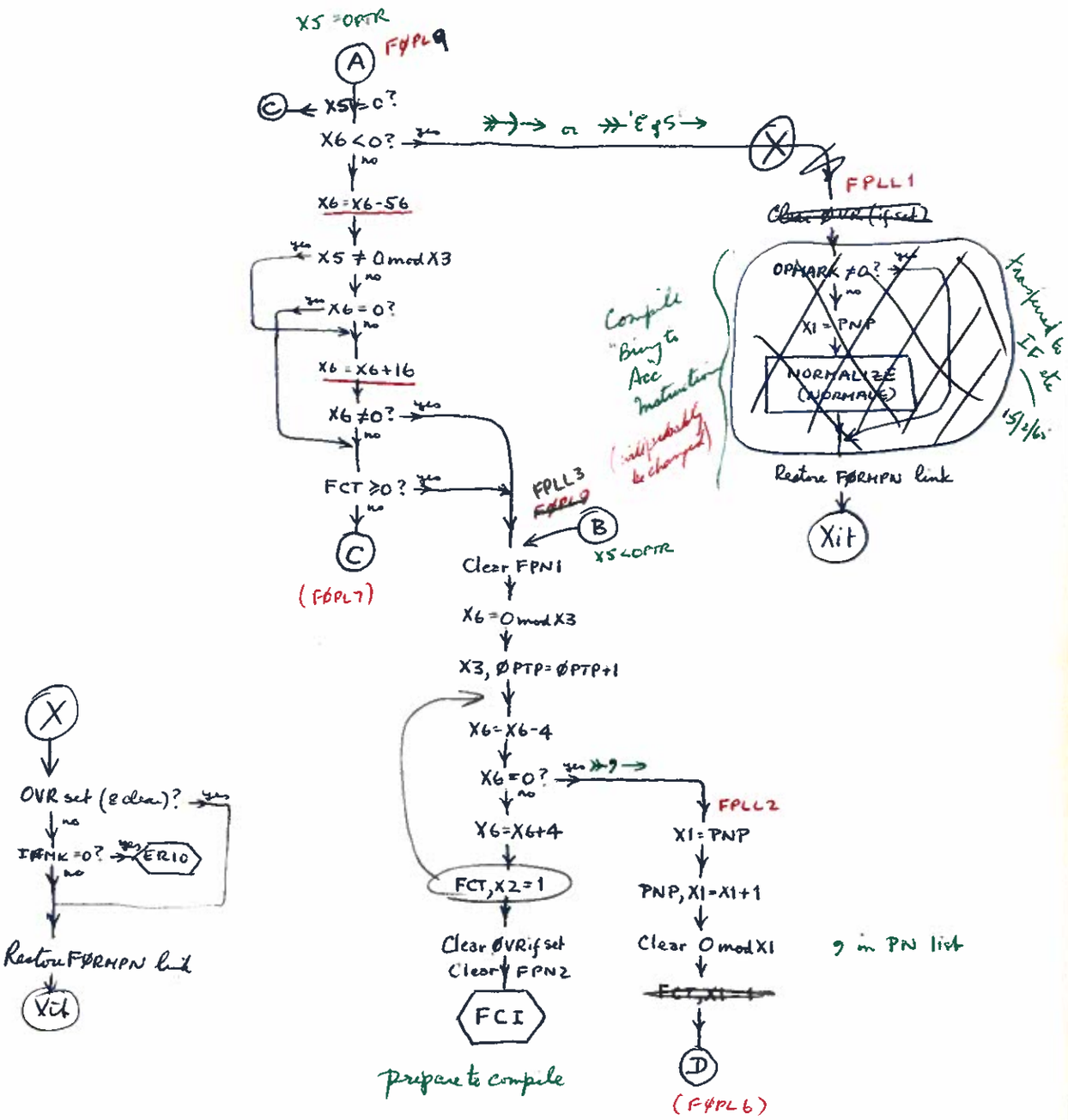
SP #204, #4000000

Genspace generates the space quantum depending on the mode of the item. For Integer or Logical, the quantum is defined by the switch setting of register 30.

3/2/65

FM


FOPE
Clear FPN4
X5 = TERM
FØPL1
X3 = 403/0 (= 3:19/0)
FØPL1
X4 = FØPT cmod X3
X4 = 0 ?
X5 = X4 ?
incr X3
X3 ≠ 0 ?
X5 = FØPT mod X3
X5 = X5 * 2^-6
Check type of Operator
FØPL2
X5 ≠ "EqS" ?
ERIO
Set ØVR
X5 = -1
Set Operator to "EqS".
FØPL3
X4 = X5
X3 = ØPTP
X4 = 0 ?
FØPL4
X4 = 1
BRACCT = BRACCT + X4
FPN2 = X4
FPN3 ≥ 0 ?
FPN3 = FPN3 - 1
C
FØPL7
X3 = X3 - 1
X3 = PNP ?
ERI3
0 mod X3 = X5
X5, FPN1 = X5
Operator into OPTR list
FØPL8
OPTP = X3
Reset Op. list pointer
FPNA
Get next field.
X6 = FPN2 *
X6 = 0 ?
X6 = FPN1
X6 ≠ 0 ?
ERIO
+ after (, = or)
FØPL5
FPNA
X4 = 40 ?
ERIO
X4 ≠ 44 ?
X5 = 48
X4 ≠ 2 ? or 4 ?
FPN2 = 1
X4 = 2 or 4
FØPL6
D
X4 = 0 mod X3
X6 = X5 v 7
X4 = (X4 U 7) - X6
X4 < 0 ?
X5 > OPTR
X6 = 0 mod X3
X4 = 0 ?
X5 = OPTR
A
X6 ≠ 0 ?
OPTR not (
B
X3 = X3 + 1
Delete (
OVR set ?
"EqS"
X4 = BRACCT
X4 = X4 - 1
ERIO
X4 < 0 ?
BRACCT = X4
X4 = 0 mod X3
X4 = 60 ?
B
"IF" operator
X4 = X5
X4 = 4 ?
X4 = BRACCT
X4 = 0 ?
X4 = IOAM
IOAM = 0
X4 = 0 ?
FPLL1
FPLL4


X5 = OPTR
FØPL9
A
X5 = 0?
X6 < 0?
yes
no
X6 = X6-56
X5 ≠ 0 mod X3
X6 = 0?
X6 = X6+16
X6 ≠ 0?
FCT ≥ 0?
C
(FØPL7)
FPLL3
FØPL9
B
X5 < OPTR
Clear FPN1
X6 = 0 mod X3
X3, ØPTP = ØPTP+1
X6 = X6-4
X6 = 0?
X6 = X6+4
FCT, X2 = 1
Clear ØVR if set
Clear FPN2
FCI
prepare to compile
FPLL2
X1 = PNP
PNP, X1 = X1+1
Clear 0 mod X1
FCT, X1 = 1
D
(FØPL6)
9 in PN list
X
FPLL1
Clear ØVR (if set)
ØPMARK ≠ 0?
X1 = PNP
NORMALIZE (NORMAL)
Restore FØRMPN link
Xit
Compile "Bring to Acc" instruction
(will probably be changed)
transfer to IF etc
15/2/65
X
OVR set (& clear)?
IFMK = 0?
ER10
Restore FØRMPN link
Xit

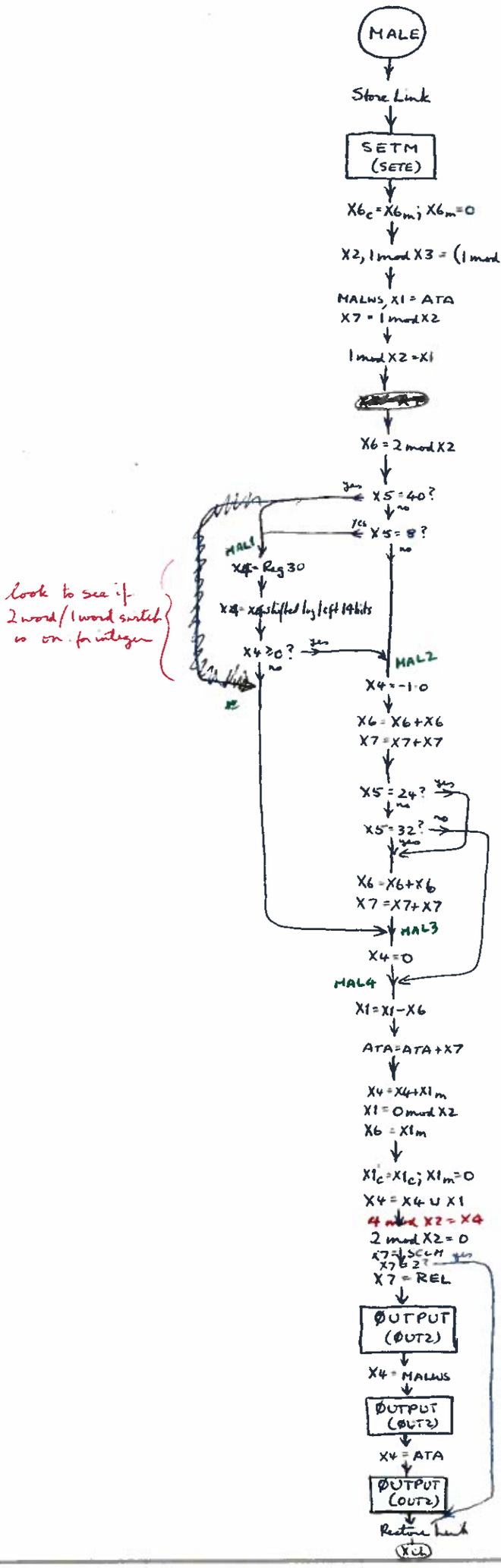
FORTRAN IV

12/2/65

FM

MALI

171



Set mode bits for Stack Item

Form Array Link item (space assigned)
(X2m = addr of array Prog)

X1 = TA for 1st element
X7 = no. of elements in array

X6 = basic offset.

Branch if logical or Integer array

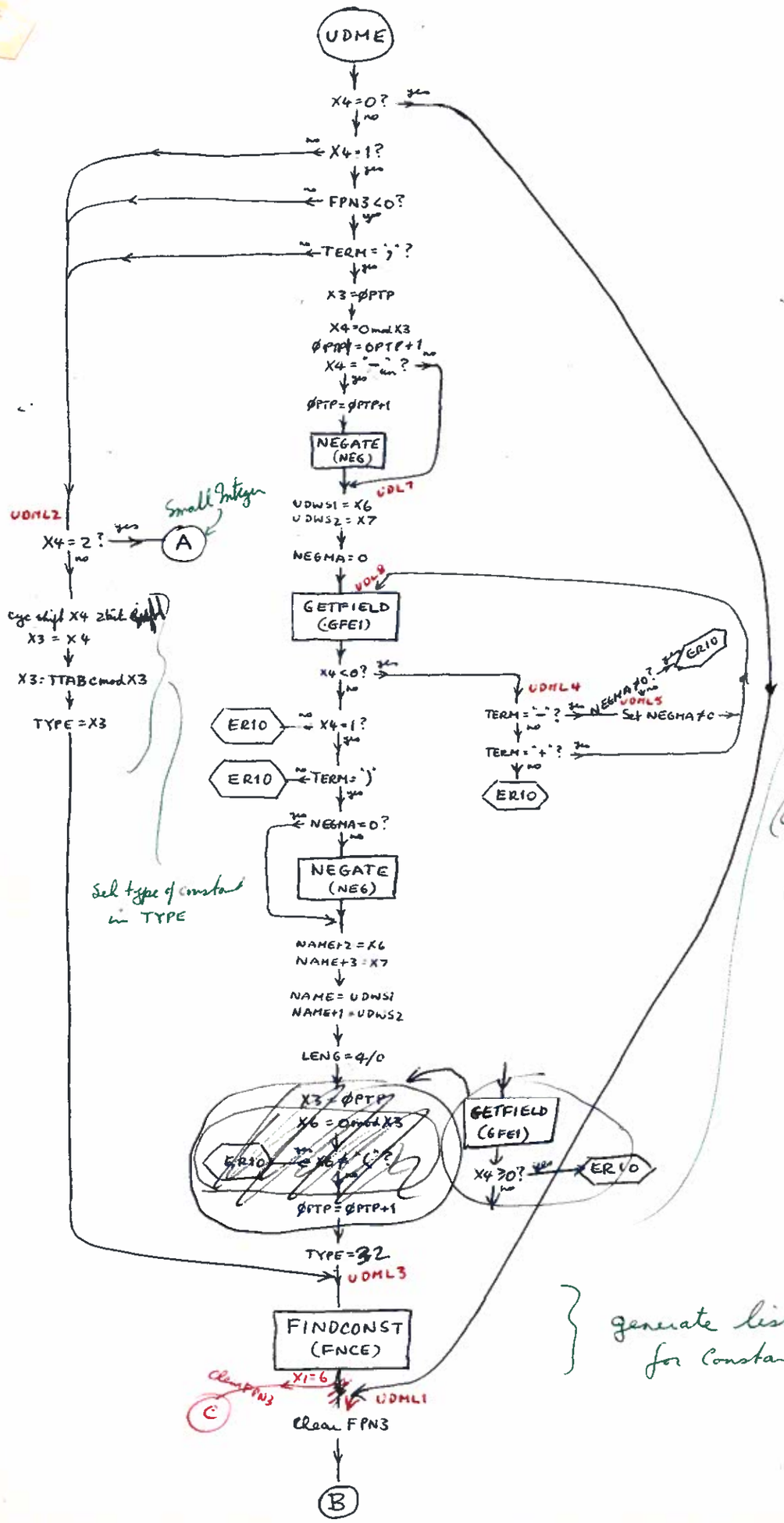
look to see if 2 word / 1 word switch is on for integer

X6 = absolute offset
X7 = Size of array
(X1 = base address)
(X4 = -1.0 for 2 word item)
ATA = next available array location

X6 = address of header

X4 = No. of dimensions / base address (+ -1.0 if 2 word item)

Output remainder of Arg Header (1st 3 words)

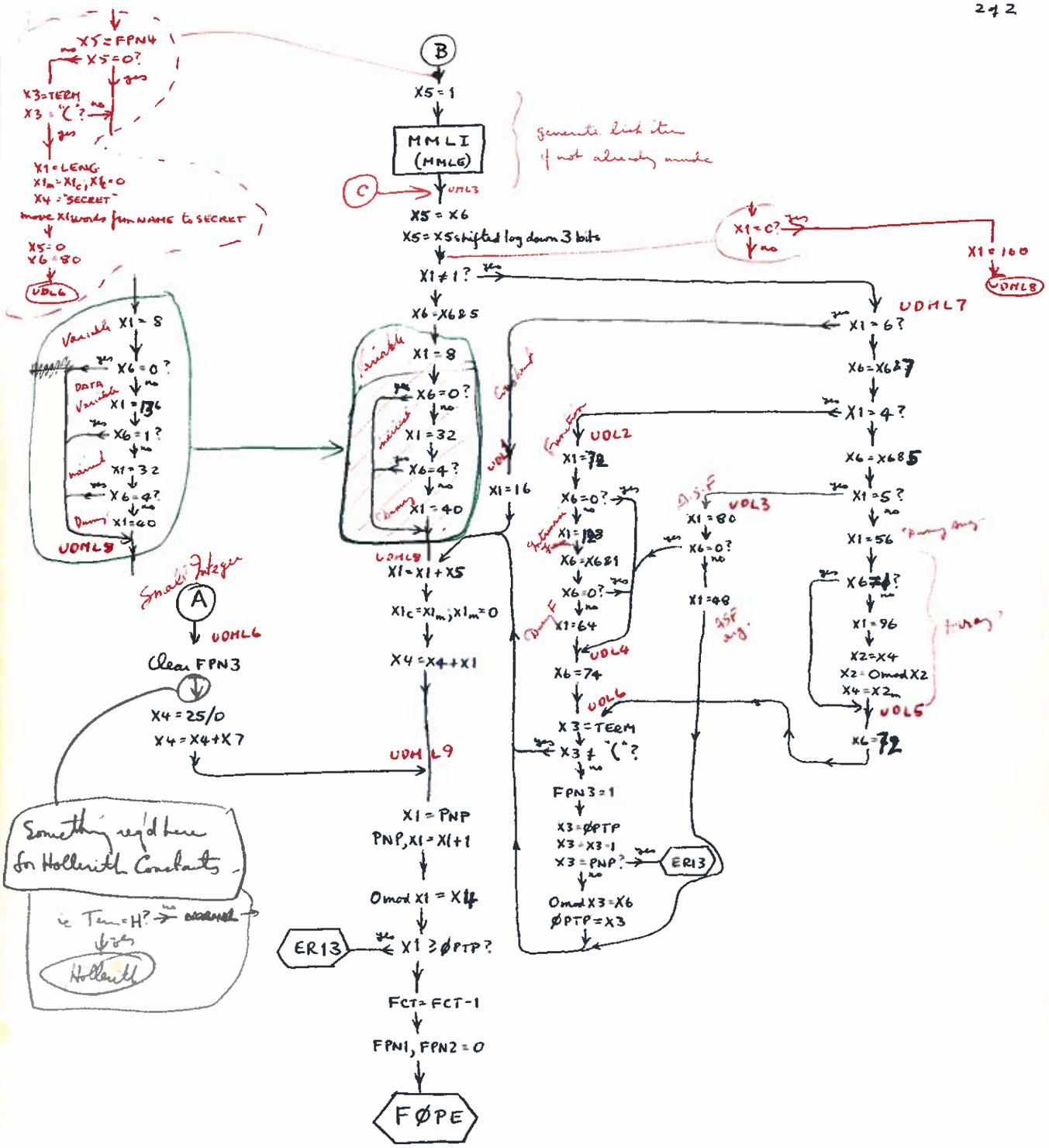


Deals with Complex Constant.

Forms it as 4 words in NAME → NAME+3

(Constant-1 form ([Sign] fl. const, [Sign] fl. const)

} generate list item for constant.



TTAB

X4 = 0
X4 = 1
X4 = 2
X4 = 3
X4 = 4

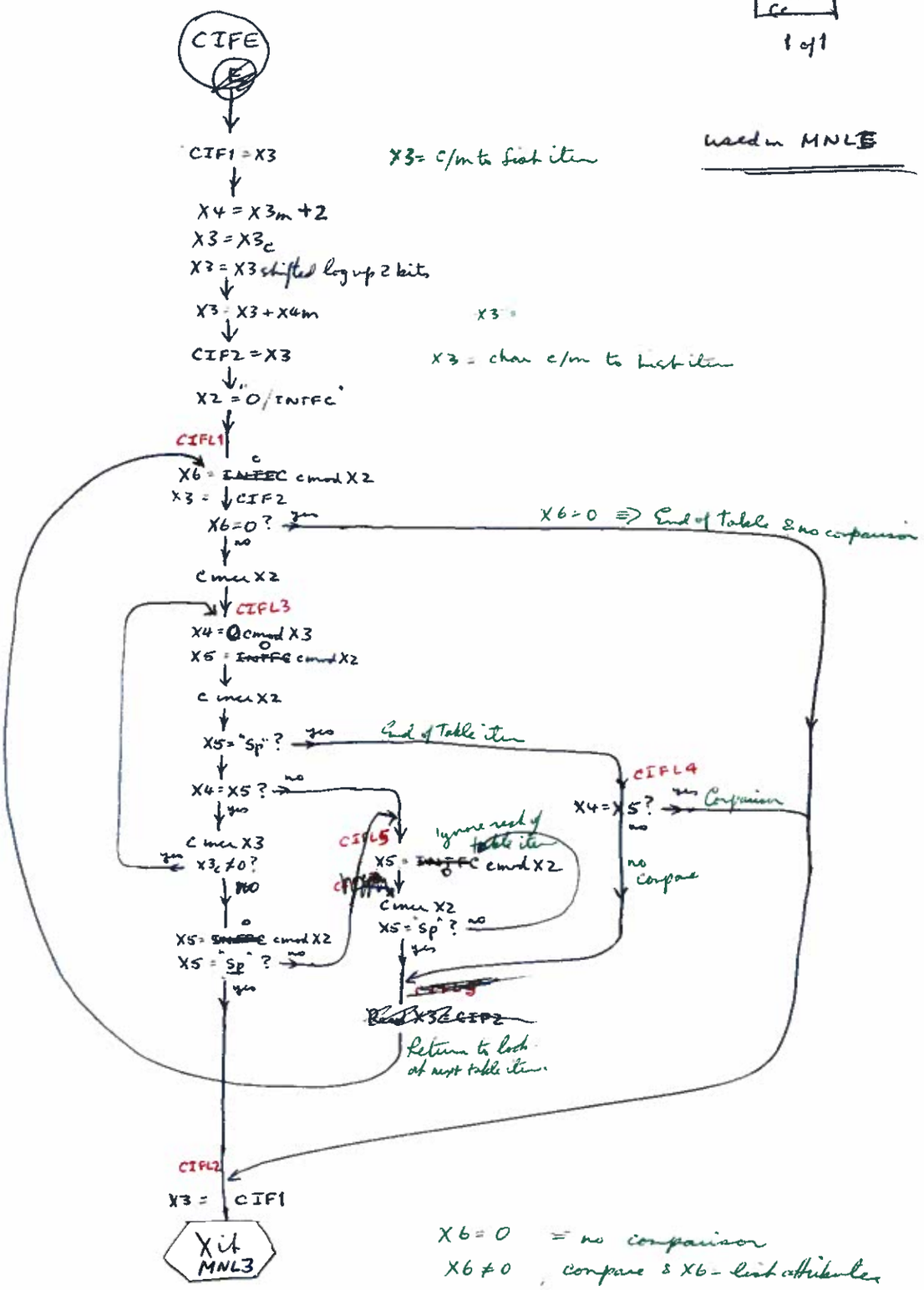
fl pt
Integer
D.P
Log.

2	0	1	3
5	0	0	0

15/2/65

FORTRAN IV

CIF
cc
1 of 1



used in MNLB

INTFC table. consists of entries as follows. 1st char. - ls. 6 bits of last attribute
2nd & succeeding chars. Function name
last char. Sp

Final Entry is terminated by zero char.

INTFC is a new Segment /INTFC

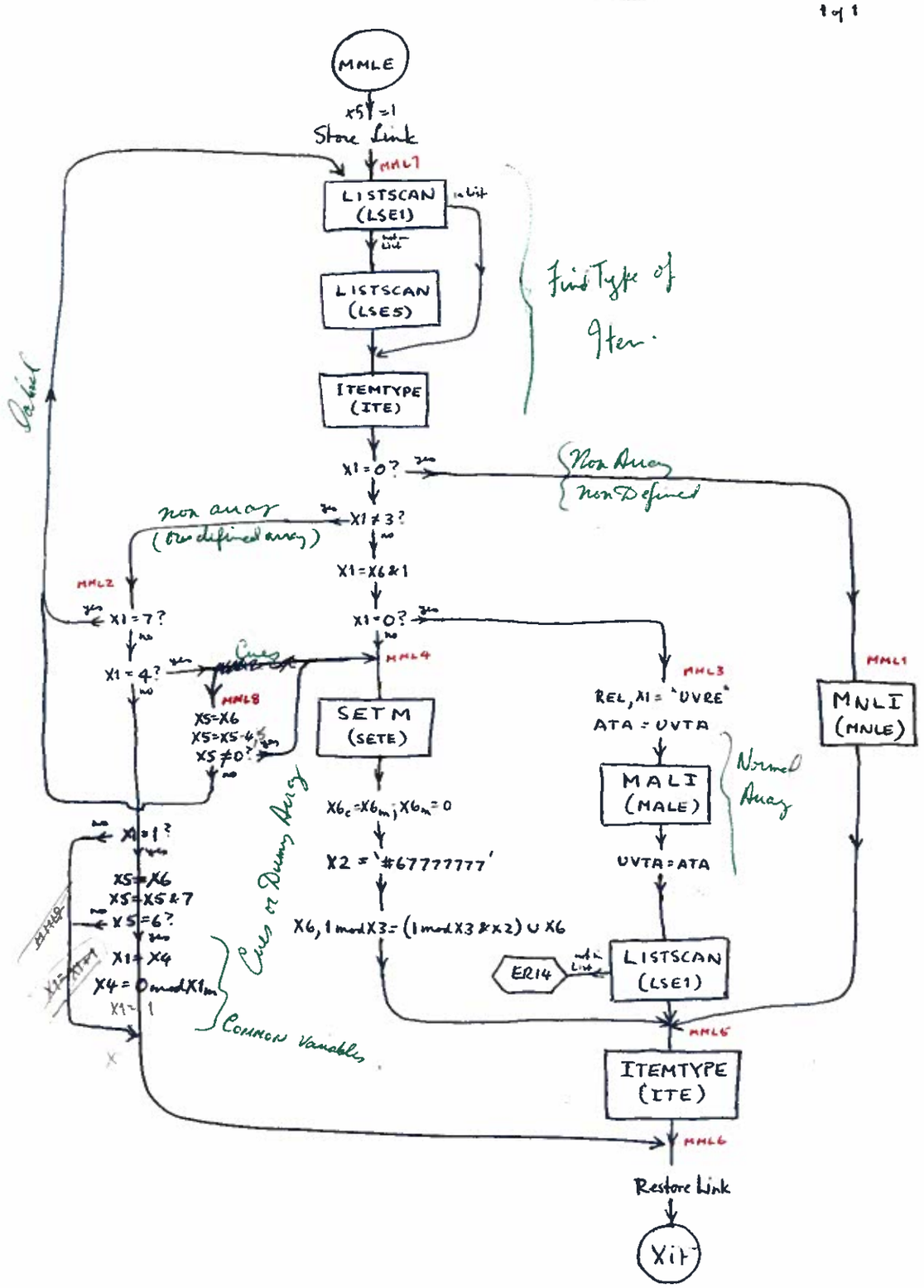
18/2/65

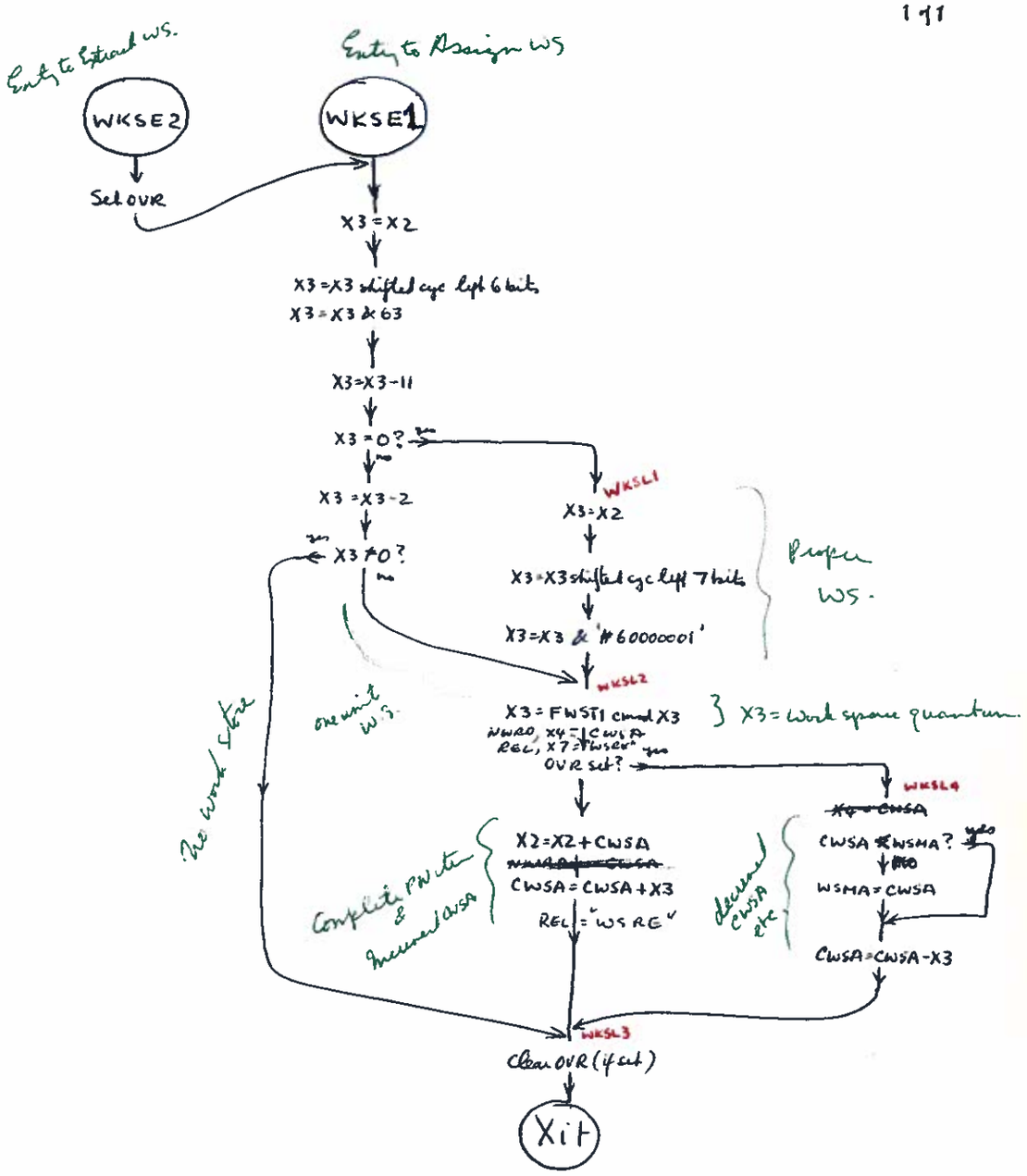
FØRTRAN IV

FM

MMLI

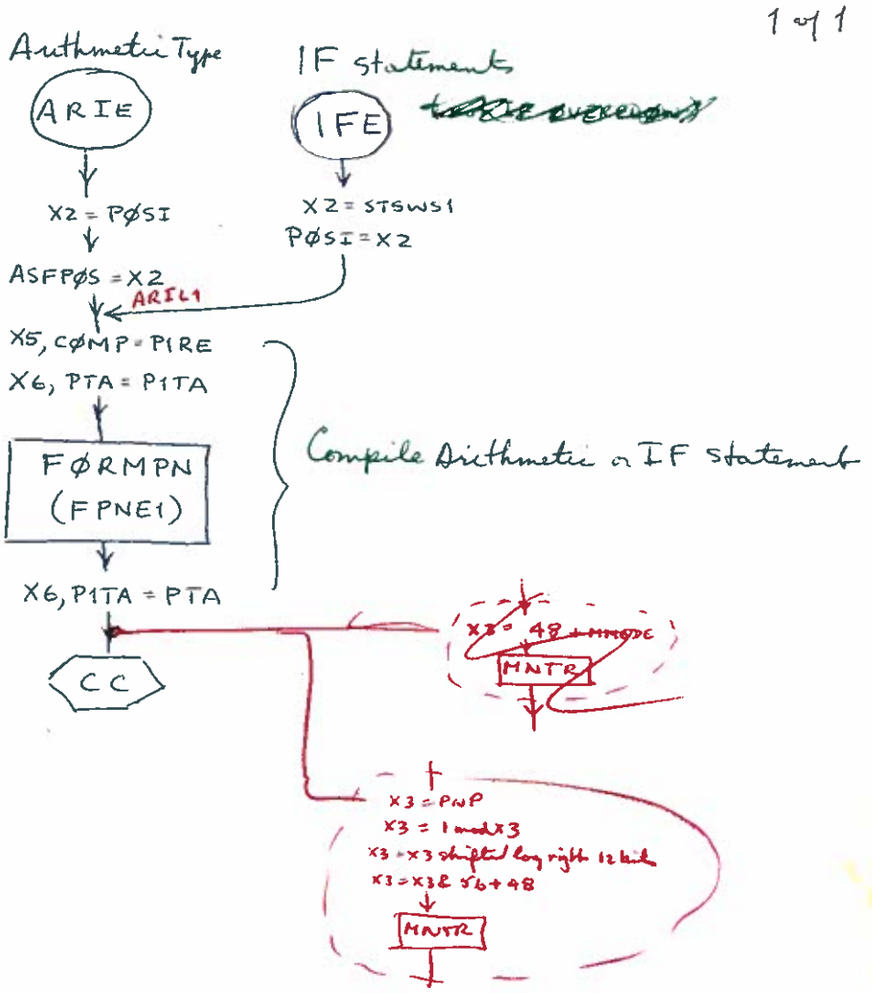
1 of 1





FWST1 #01010204, #04010000

Entry Points	WKSE1	Enter to assign Work space if required.	Increments CWSA & completes PN item in X2
	WKSE2	Enter to extend WS if need.	Decrement CWSA & resets WSMA.



Note STSWS1 is the work store of STS

8/3/65

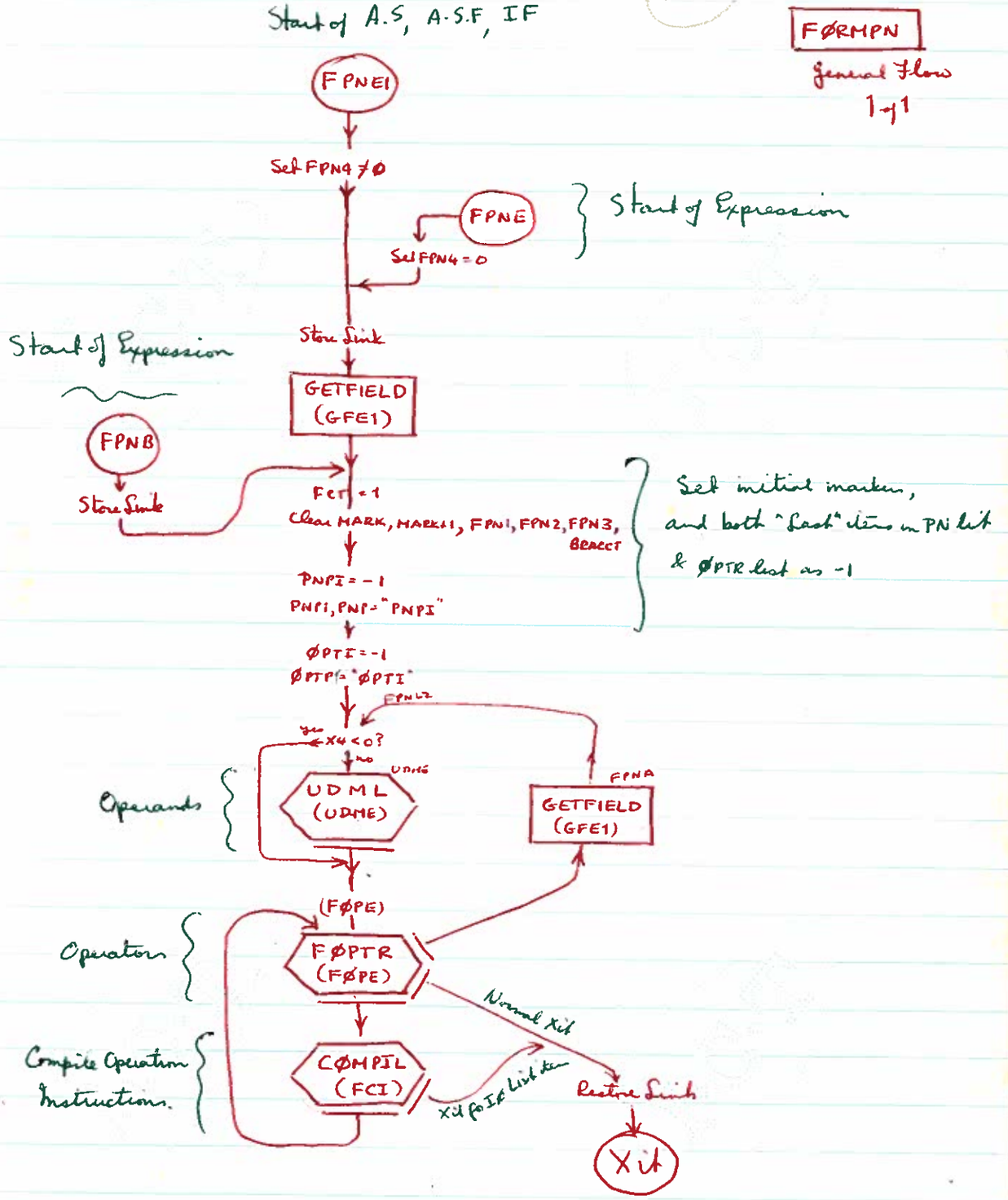
FORTRAN IV

FM

FØRMPN

General Flow

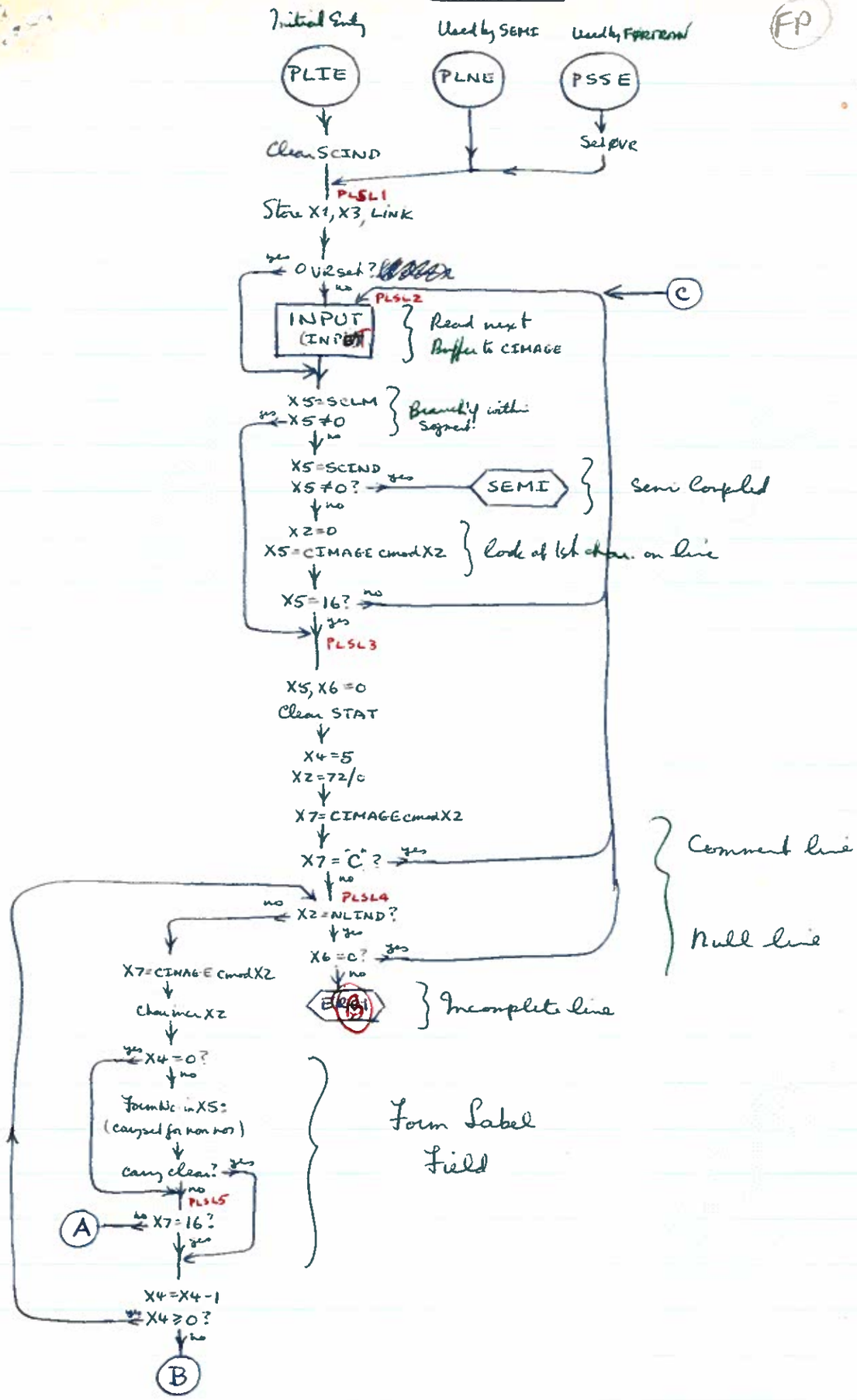
1 of 1



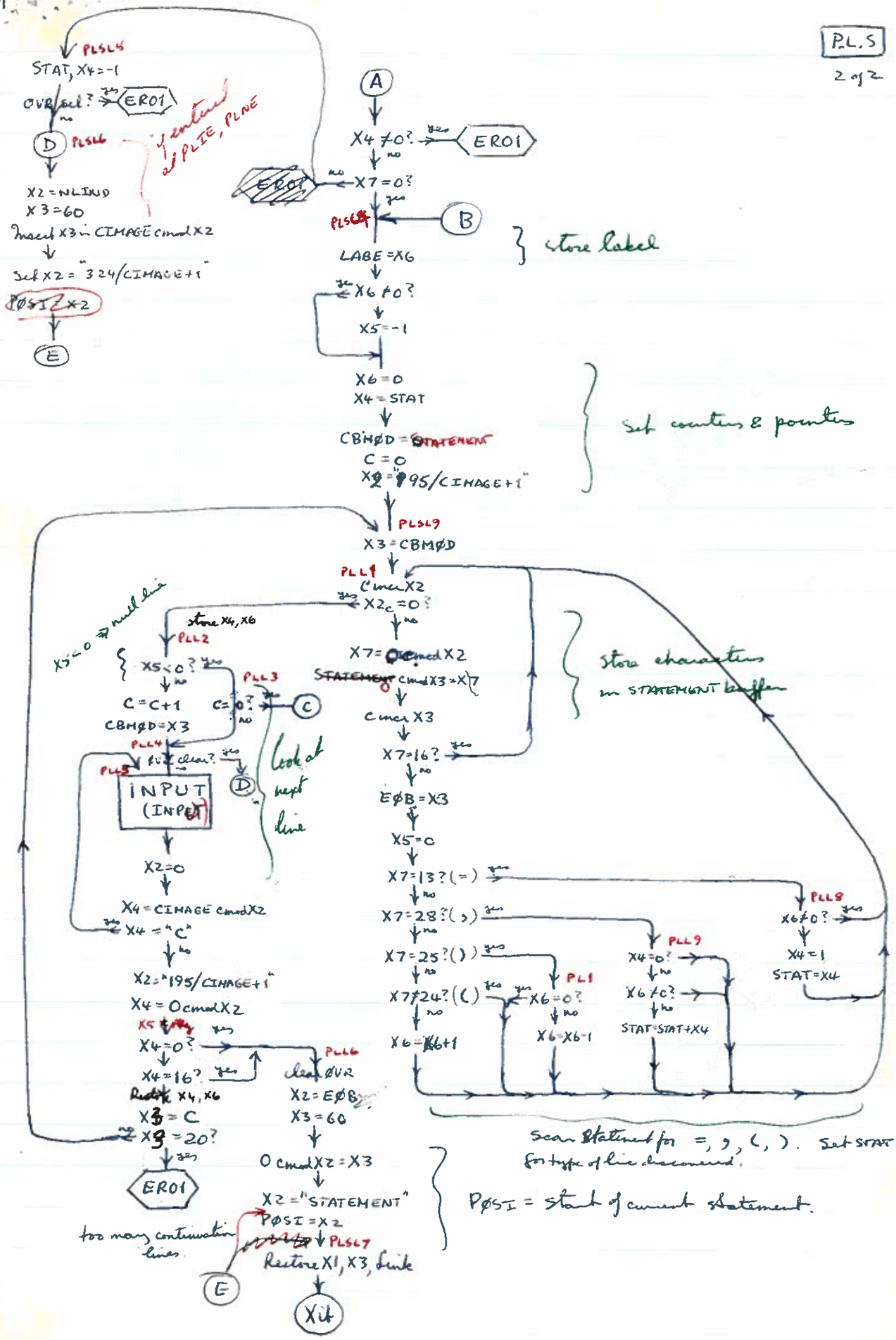
FP

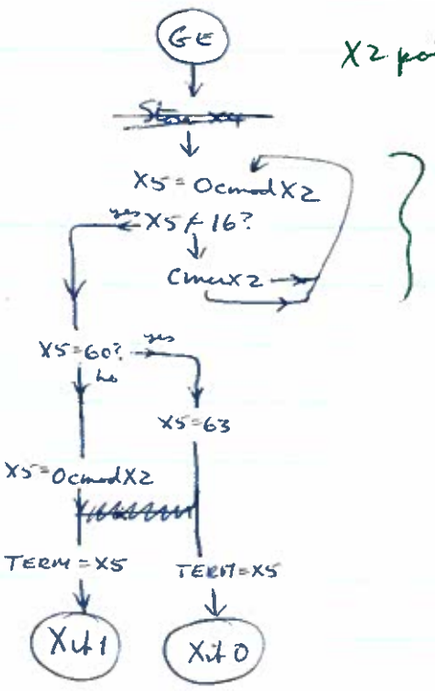
P.L.S.

1 of 2


Initial Entry
Used by SEMI
Used by FORTRAN
PLIE
PLNE
PSSE
Clear SCIND
Set ovr
PLSL1
Store X1, X3, LINK
ovr set?
PLSL2
INPUT
Read next Buffer to CIMAGE
X5=SCLM
X5≠0
Branch if within Segment
X5=SCIND
X5≠0?
SEMI
Semi Compiled
X2=0
X5=CIMAGE cmod X2
Code of 1st char. on line
X5=16?
PLSL3
X5, X6=0
Clear STAT
X4=5
X2=72/6
X7=CIMAGE cmod X2
X7='C'?
Comment line
PLSL4
X2=NLIND?
X6=6?
Null line
X7=CIMAGE cmod X2
Char in X2
Incomplete line
X4=0?
Form Label Field
Form bit in X5: (caused for non nos)
carry clear?
PLSL5
A
X7=16?
X4=X4-1
X4≥0?
B

FØRTRAN IV


PLSL4
STAT, X4=-1
OVR set? yes ERO1
no
D PLSL6
X2=NLIND
X3=60
Insert X3 in CIMAGE cmod X2
Set X2 = "324/CIMAGE+1"
PØSI=X2
E
I entered at PLIE, PLNE
A
X4≠0? yes ERO1
no
X7=0? no ERO1
yes
PLSL5
B
LABE=X6
X6≠0? yes
X5=-1
X6=0
X4=STAT
CBMØD=STATEMENT
C=0
X2="195/CIMAGE+1"
store label
Set counters & pointers
PLSL9
X3=CBMØD
PLL1
C cmod X2
X2c=0? yes
no
X7=C cmod X2
STATEMENT C cmod X3=X7
C cmod X3
X7=16? yes
no
EØB=X3
X5=0
store characters in STATEMENT buffer
X5<0? yes
no
store X4, X6
X5=0 ⇒ null line
PLL2
C=C+1
C=0? yes
no
PLL3
C
CBMØD=X3
PLL4
ØVR clear? yes
no
PLL5
INPUT (INPEX)
D
Look at next line
X2=0
X4=CIMAGE cmod X2
X4="C" yes
no
X2="195/CIMAGE+1"
X4=0 cmod X2
X5=0
X4=0? yes
no
X4=16? yes
no
Restore X4, X6
X3=C
X3=20? yes
no
ERO1
too many continuation lines
E
PLL6
clear ØVR
X2=EØB
X3=60
0 cmod X2=X3
X2="STATEMENT"
PØSI=X2
PLSL7
Restore X1, X3, link
Exit
X7=13? (=) yes
no
X7=28? (>) yes
no
X7=25? ()) yes
no
X7=24? (() yes
no
X6=X6+1
PL1
X6=0? yes
no
X6=X6-1
PLL9
X4=0? yes
no
X6≠0? yes
no
STAT=STAT+X4
PLL8
X6≠0? yes
no
X4=1
STAT=X4
Scan Statement for =,), (,). Set STAT for type of line encountered.
PØSI = start of current statement.



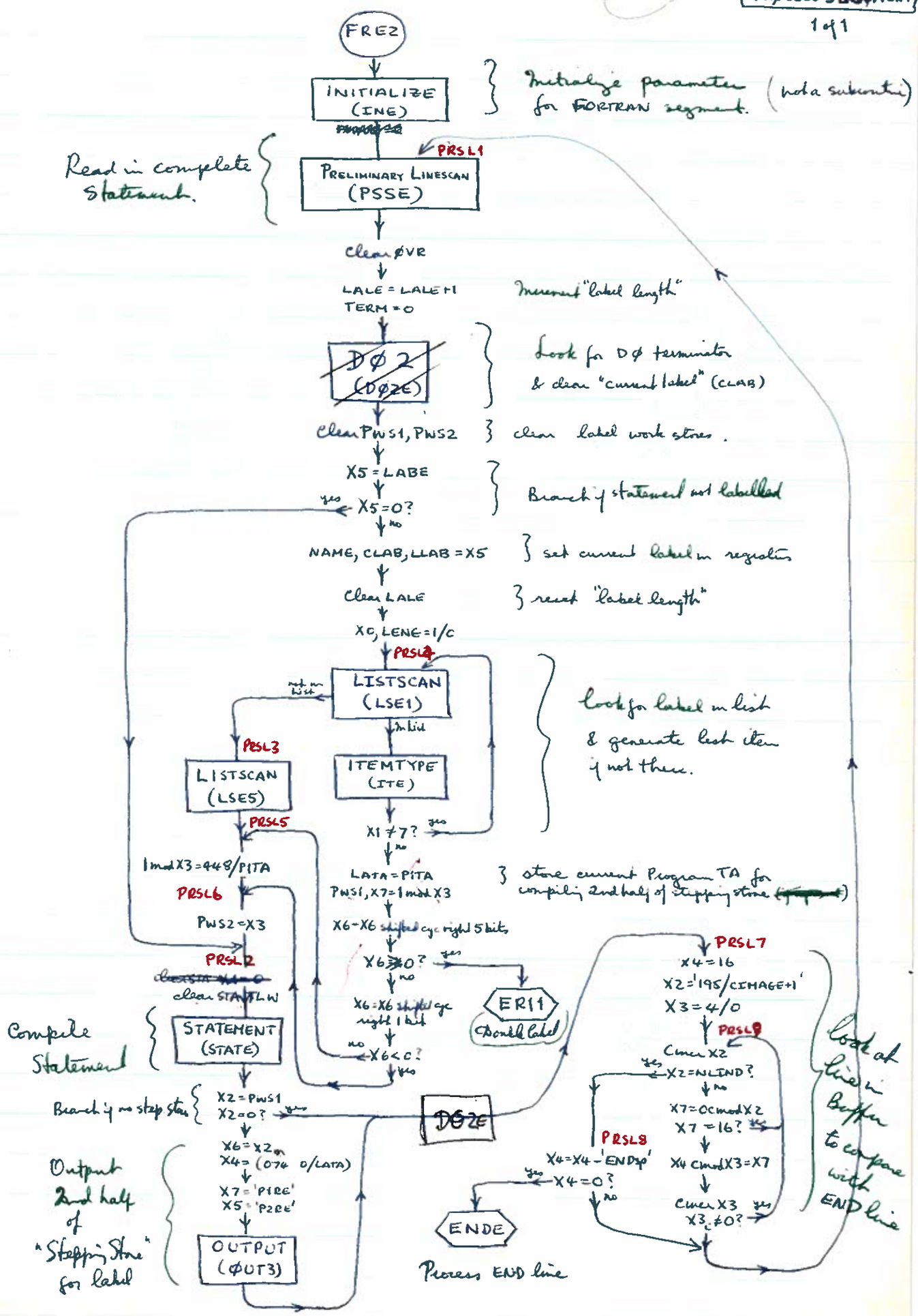
X2 points at char

look for 1st non space char.

X2 point at char found.

Routine used to get every character found by Getfield

Exit 1. X5 contains 1st non-space char. } also TERM.
0 X5 contains Eq 5 char. }


FRE2
INITIALIZE (INE)
Initialize parameter for FORTRAN segment. (not a subroutine)
PRSL1
Read in complete Statement.
PRELIMINARY LINESCAN (PSSE)
Clear ØVR
LALE = LALE+1
TERM = 0
Increment "label length"
DØ 2 (DØ2E)
Look for DØ terminator & clear "current label" (CLAB)
Clear PWS1, PWS2
clear label work stores.
X5 = LABE
X5=0?
Branch if statement not labelled
NAME, CLAB, LLAB = X5
set current label in registers
Clear LALE
reset "label length"
X0, LENE = 1/C
PRSL4
LISTSCAN (LSE1)
not on list
in list
Look for label in list & generate list item if not there.
PRSL3
LISTSCAN (LSE5)
ITEMTYPE (ITE)
PRSL5
X1 ≠ 7?
yes
no
1 mod X3 = 448/PITA
LATA = PITA
PWS1, X7 = 1 mod X3
store current Program TA for compiling 2nd half of Stepping stone
PRSL6
PWS2 = X3
X6 = X6 shifted cyc right 5 bits
X6 ≥ 0?
ER11
Double Label
PRSL2
clear STATEMENT
X6 = X6 shifted cyc right 1 bit
X6 < 0?
Compile Statement
STATEMENT (STATE)
Branch if no step stone
X2 = PWS1
X2 = 0?
DØ2E
PRSL7
X4 = 16
X2 = '195/CIMAGE+1'
X3 = 4/0
PRSL9
CIMEM X2
X2 = NLIND?
X7 = 0C mod X2
X7 = 16?
X4 CMOD X3 = X7
CIMEM X3
X3 ≠ 0?
Look at line in Buffer to compare with END line
PRSL8
X4 = X4 - 'END '
X4 = 0?
ENDE
Process END line
X6 = X2
X4 = (07400/LATA)
X7 = 'P1EE'
X5 = 'P2EE'
OUTPUT (ØUT3)
Output 2nd half of "Stepping Stone" for label

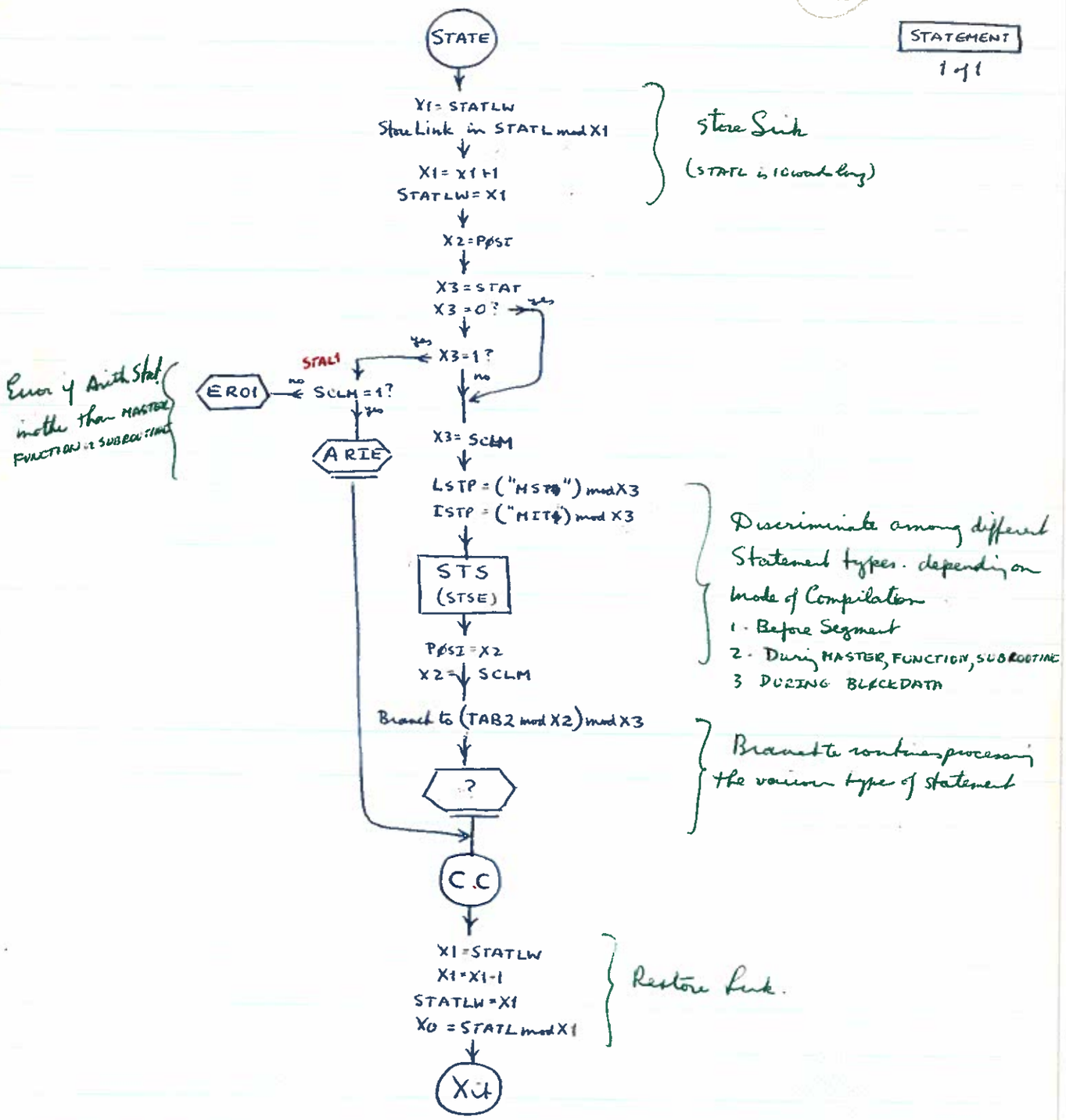
13/3/65

FORTRAN IV

FE

STATEMENT

1 of 1

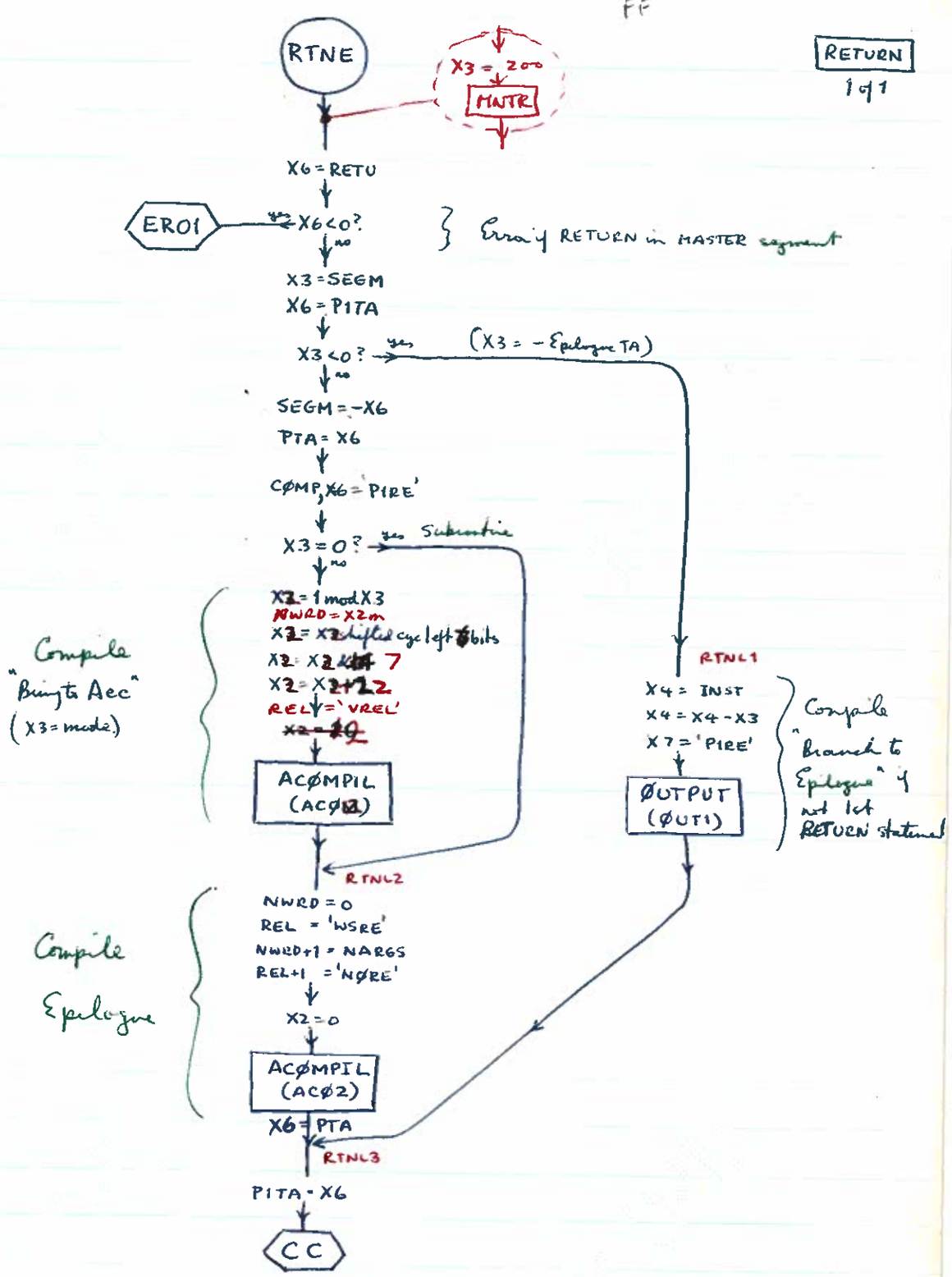


14/3/65

FF

RETURN

191



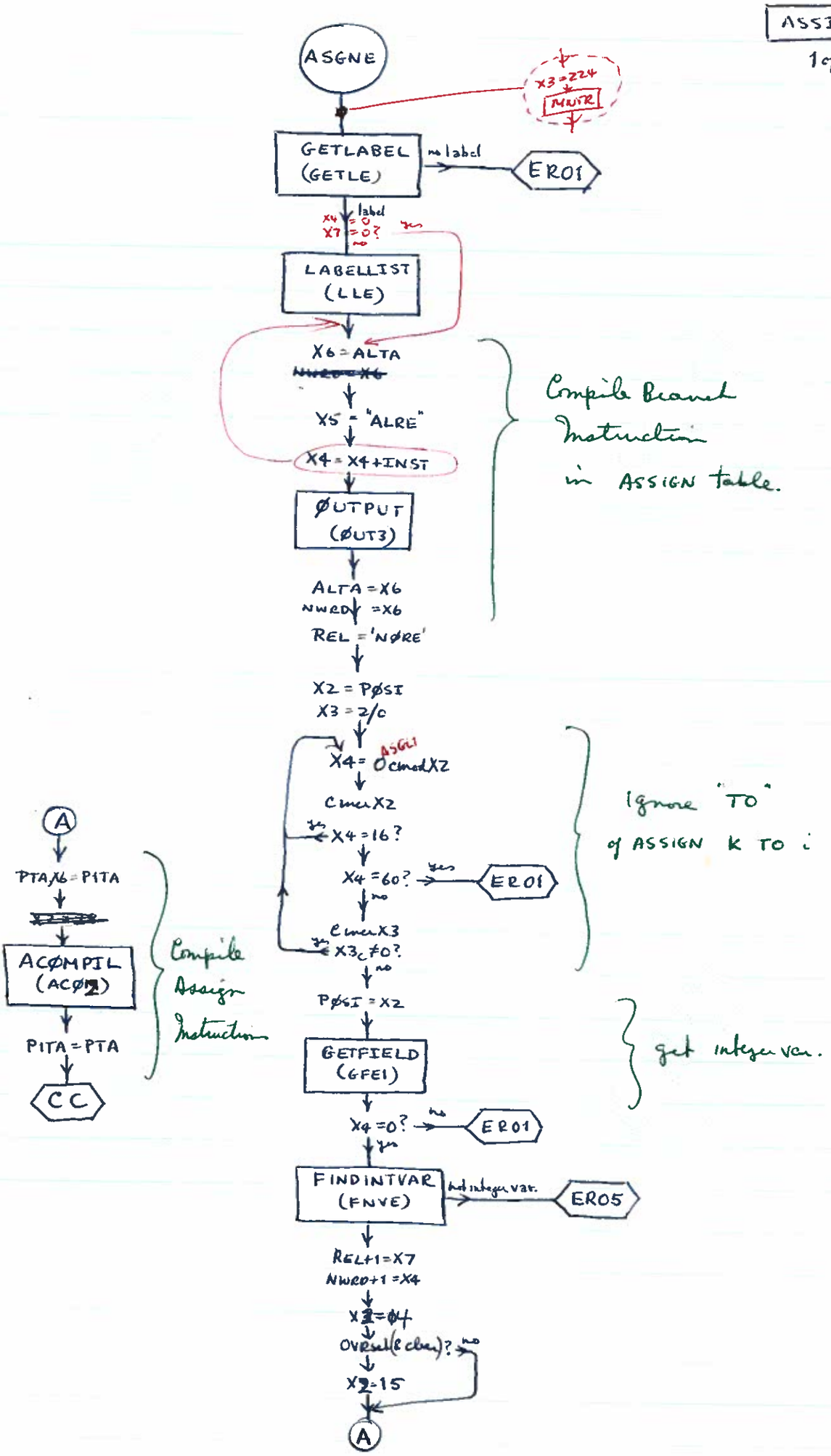
14/3/65

FØKIKAN IV

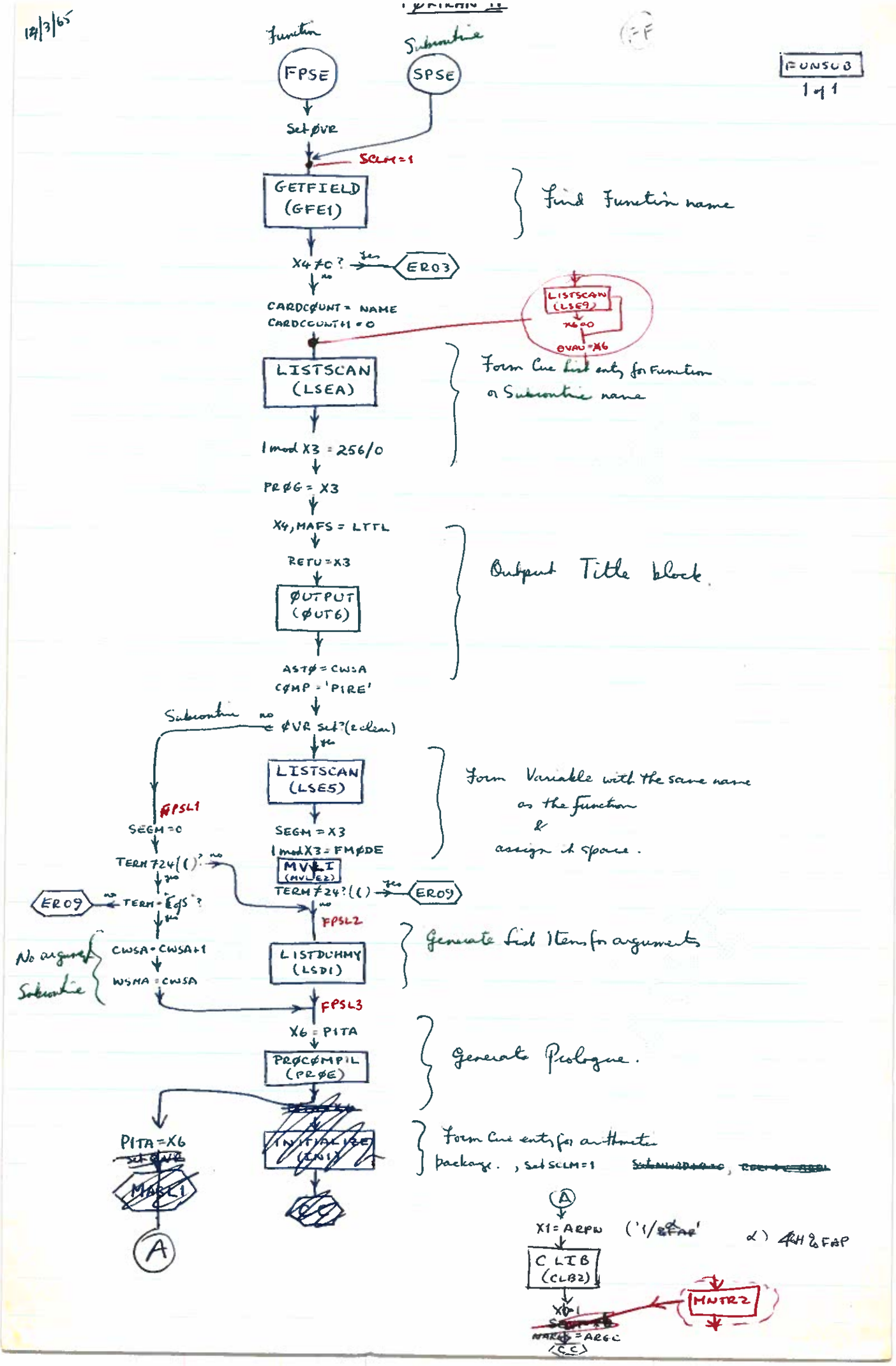
FH

ASSIGN

1 of 1



12/3/65

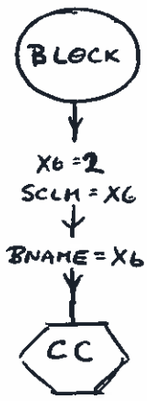

Function
FPSE
Subroutine
SPSE
Set ØVR
SCLM=1
GETFIELD (GFE1)
Find Function name
X4≠0 ?
yes
ERO3
no
CARDCØUNT = NAME
CARDCOUNTH = 0
LISTSCAN (LSE9)
X6=0
EVAL=X6
LISTSCAN (LSEA)
Form Cue list entry for Function or Subroutine name
I mod X3 = 256/0
PRØG = X3
X4, MAFS = LTTL
RETU = X3
ØUTPUT (ØUT6)
Output Title block
ASTØ = CWSA
CØMP = 'PIRE'
Subroutine
no
ØVR set? (reclean)
yes
LISTSCAN (LSE5)
Form Variable with the same name as the function & assign it space.
FPSL1
SEGM = 0
SEGM = X3
I mod X3 = FMØDE
TERM 724?((?
yes
MVLI (MVLE2)
TERM 724?((yes
ERO9
no
FPSL2
ERO9
TERM = EQS ?
yes
No argument Subroutine
CWSA = CWSA+1
WSMA = CWSA
LISTDUMMY (LSD1)
Generate list items for arguments
FPSL3
X6 = PITA
PRØCØMPIL (PRØE)
Generate Prologue.
PITA = X6
MAGIC1
INITIALISE (INI1)
Form Cue entry for arithmetic package. , set SCLM=1
A
X1 = ARPN
('1/
α) ARH & FAP
C LIB (CLB2)
MNTR2
MAIN = ARGC
(C C)

FOKIKAN IV

14/3/65

BLOCK DATA

1 of 1

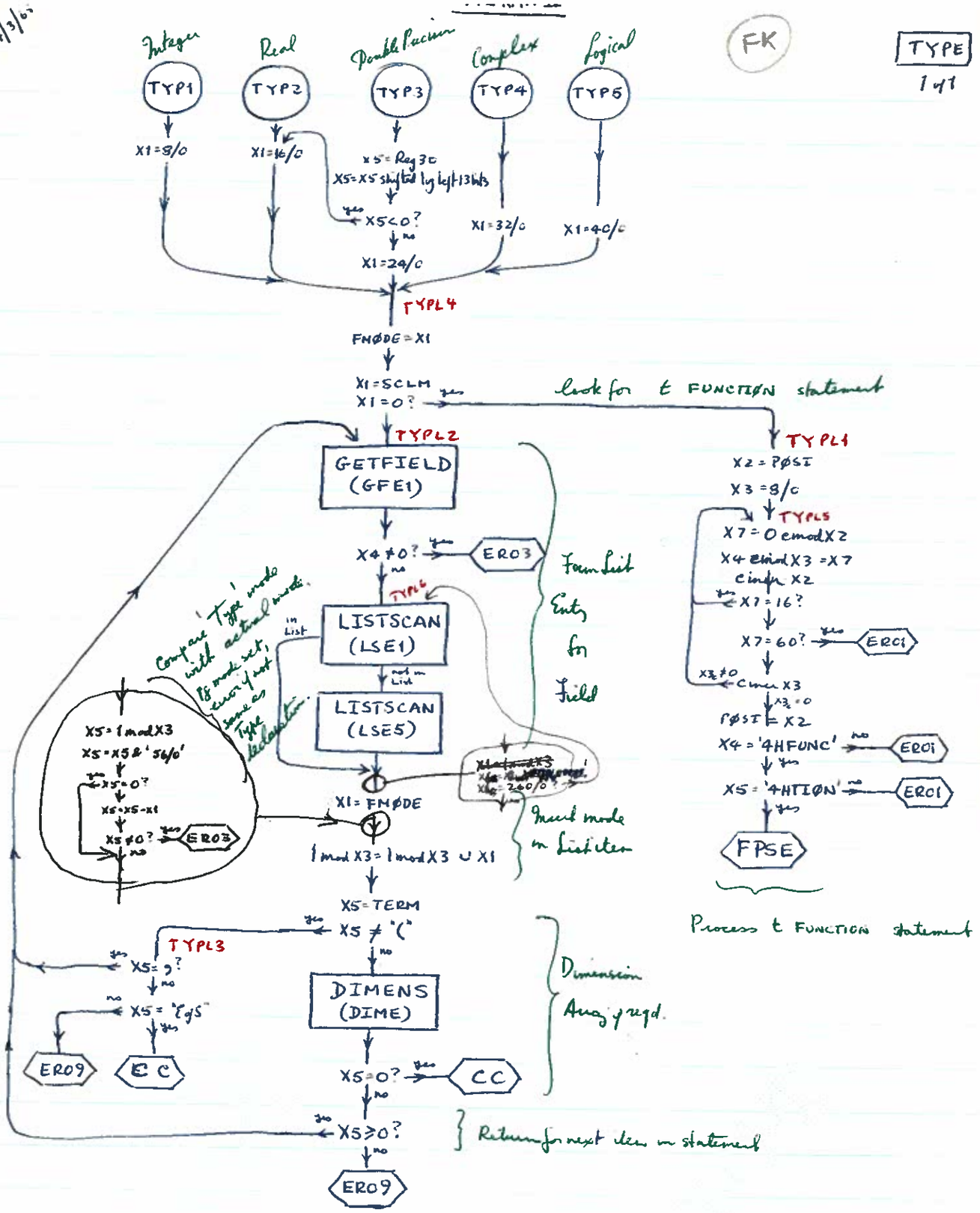


Set Block data marbo

14/3/67

FK

TYPE
1 of 1



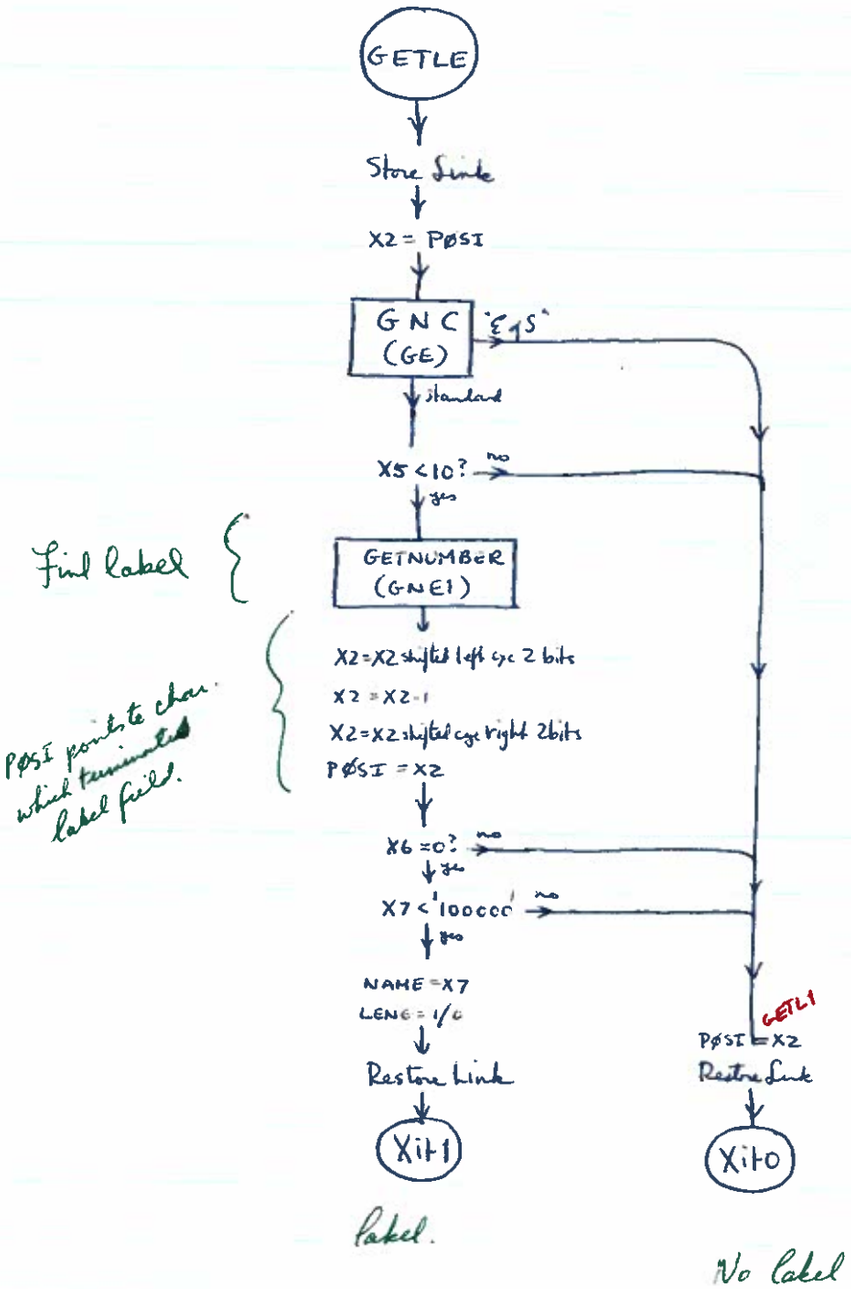
14/3/63

FØRTRAN IV

FP

GETLABEL

141

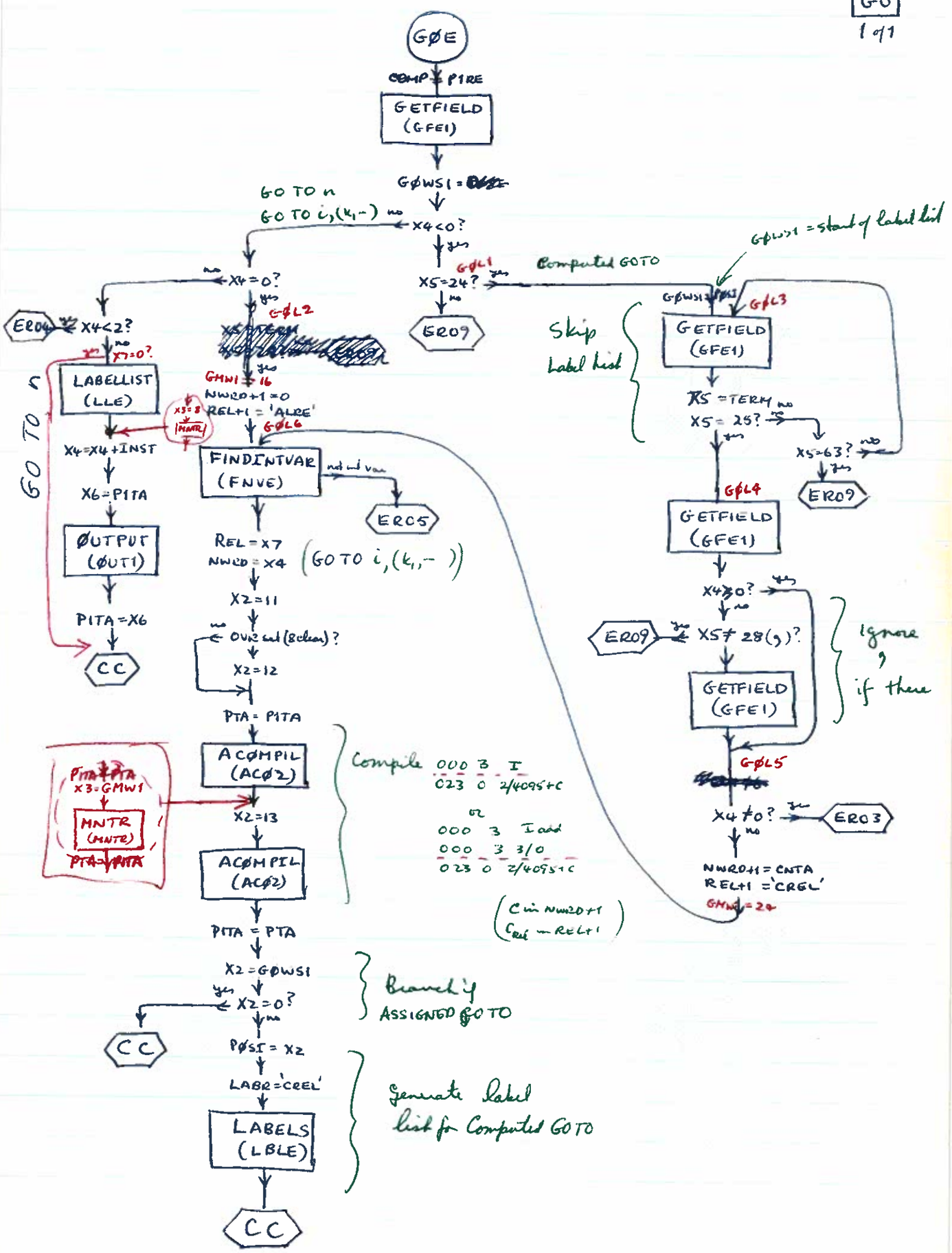


15/3/65

FORTRAN IV

FH

G-0
1 of 1

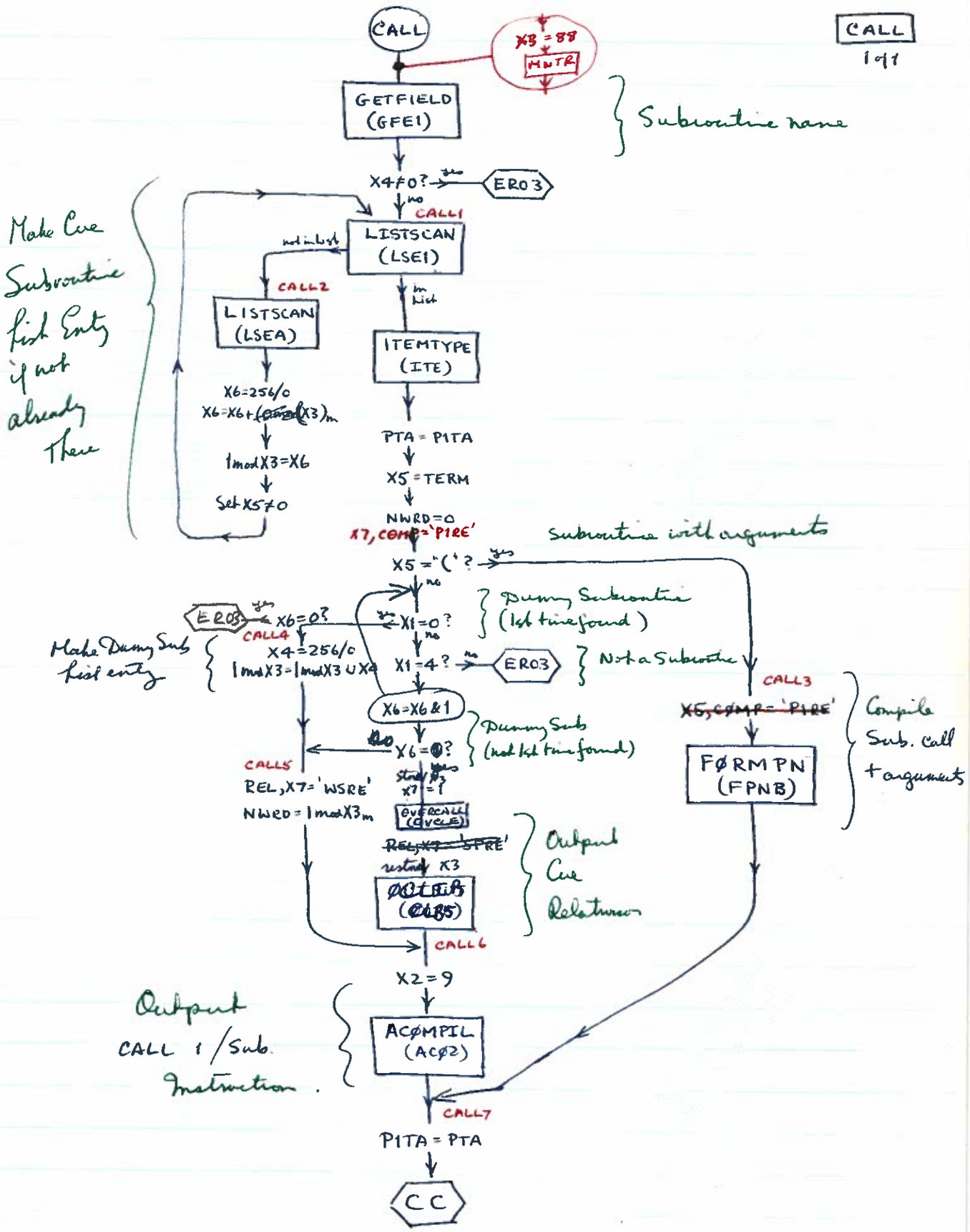


15/3/16

FH

CALL

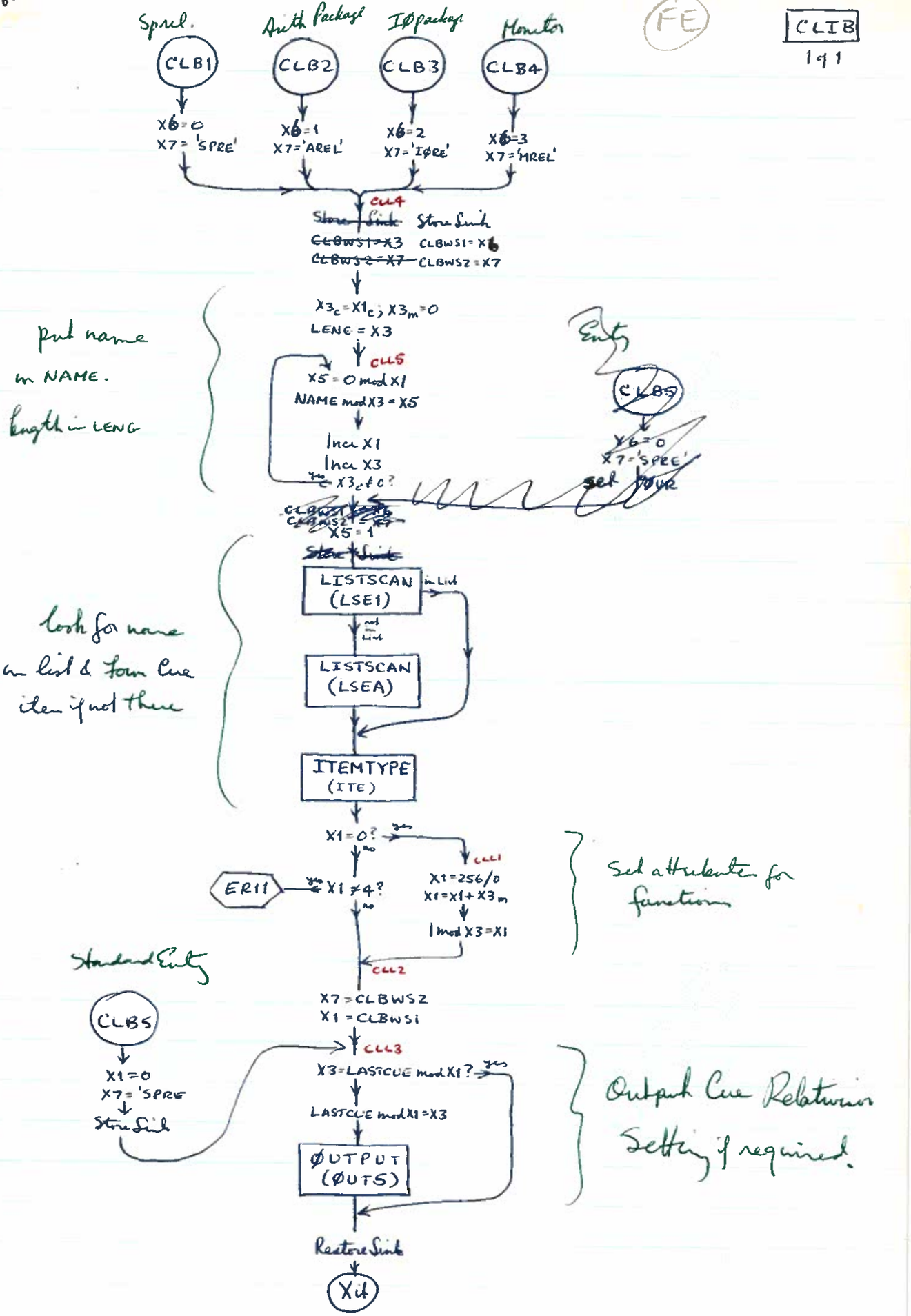
191

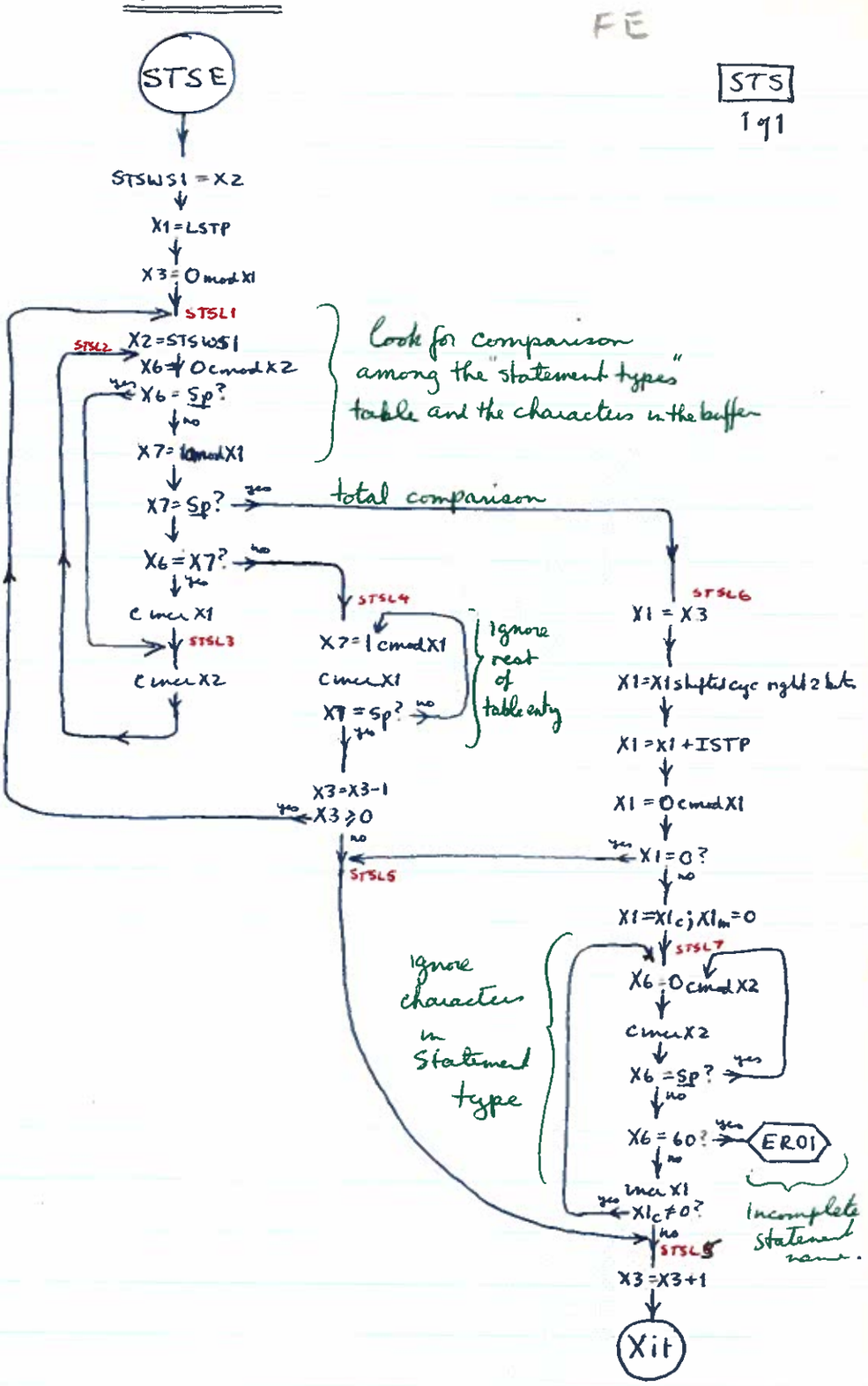


16/3/65

FE

CLIB
191



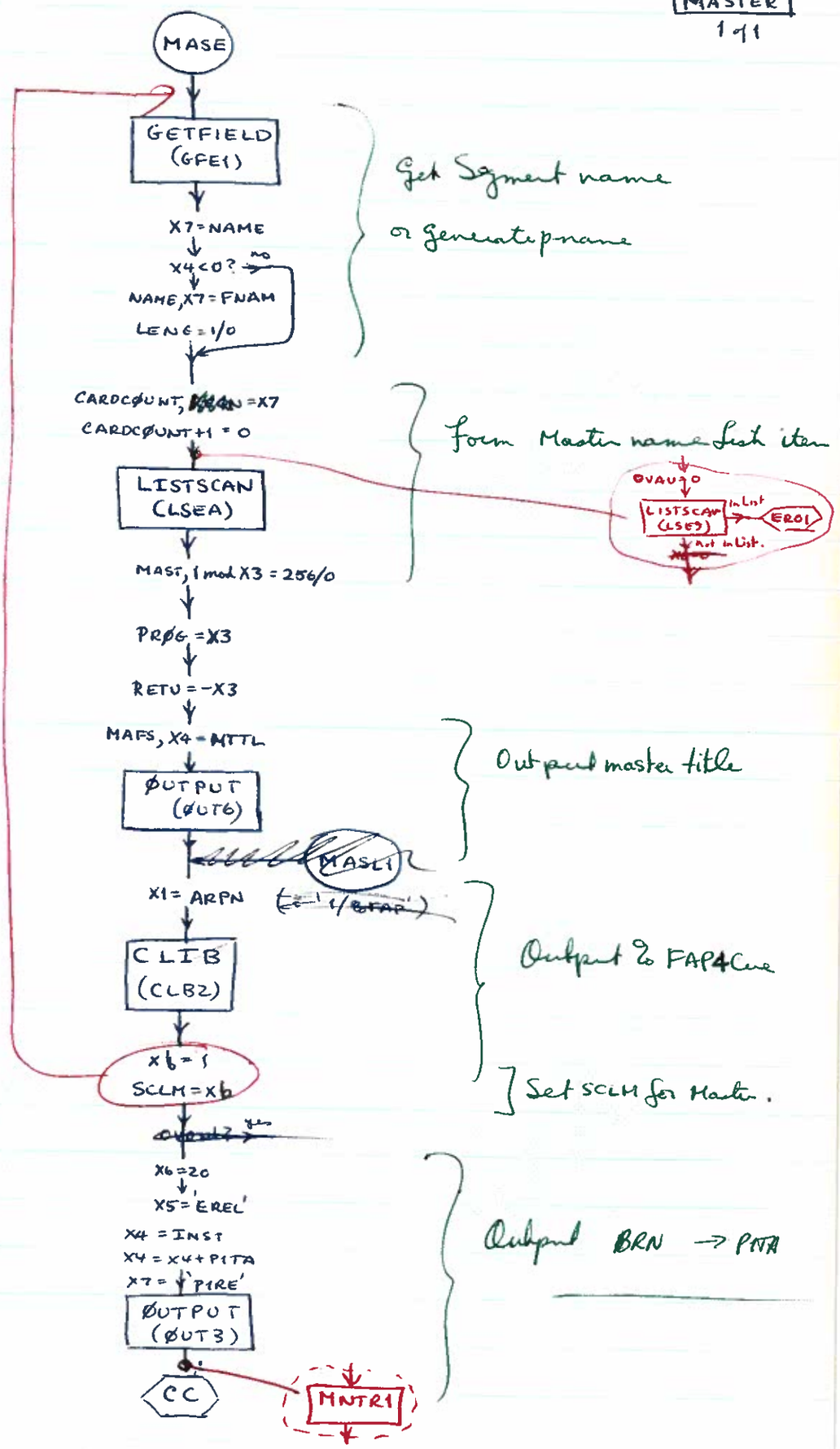

STSE
STSWS1 = X2
X1 = LSTP
X3 = 0 cmod X1
STSL1
X2 = STSWS1
STSL2
X6 = 0 cmod X2
X6 = Sp?
yes
no
X7 = 1 cmod X1
X7 = Sp?
yes
X6 = X7?
yes
no
cmcu X1
STSL3
cmcu X2
STSL4
X7 = 1 cmod X1
cmcu X1
X7 = Sp?
no
yes
Look for comparison among the "statement types" table and the characters in the buffer
total comparison
Ignore rest of table entry
STSL6
X1 = X3
X1 = X1 shifted cyc rgtd 2 bit.
X1 = X1 + ISTP
X1 = 0 cmod X1
X1 = 0?
yes
no
X1 = X1c ; X1m = 0
STSL5
X3 = X3 - 1
X3 ≥ 0
yes
no
STSL7
X6 = 0 cmod X2
cmcu X2
X6 = Sp?
yes
no
X6 = 60?
yes
ERO1
Incomplete statement name.
cmcu X1
X1c ≠ 0?
yes
no
Ignore characters in Statement type
STSL8
X3 = X3 + 1
Xit

18/3/65

FØRTRAN IV

(FF)

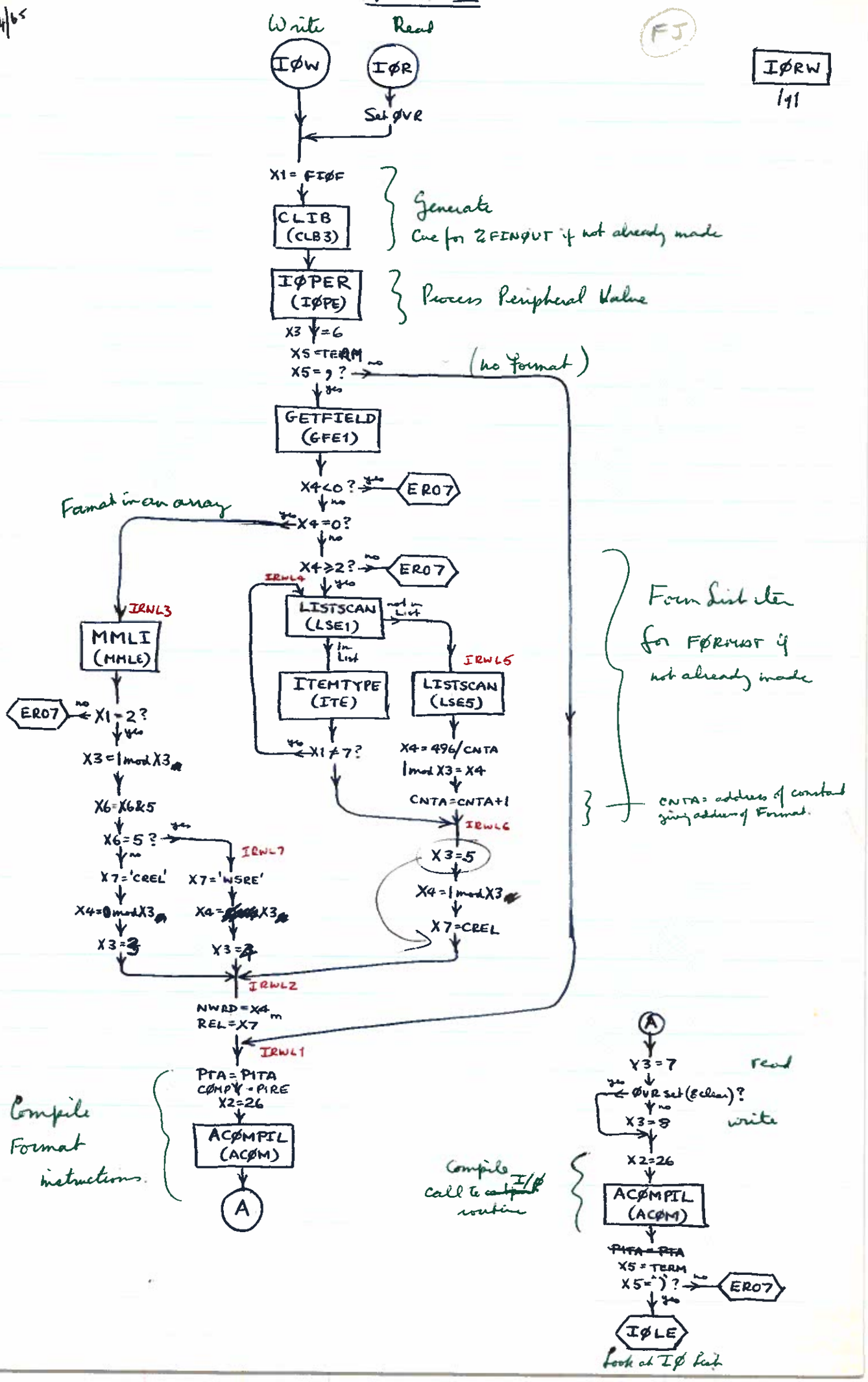
MASTER
1 of 1

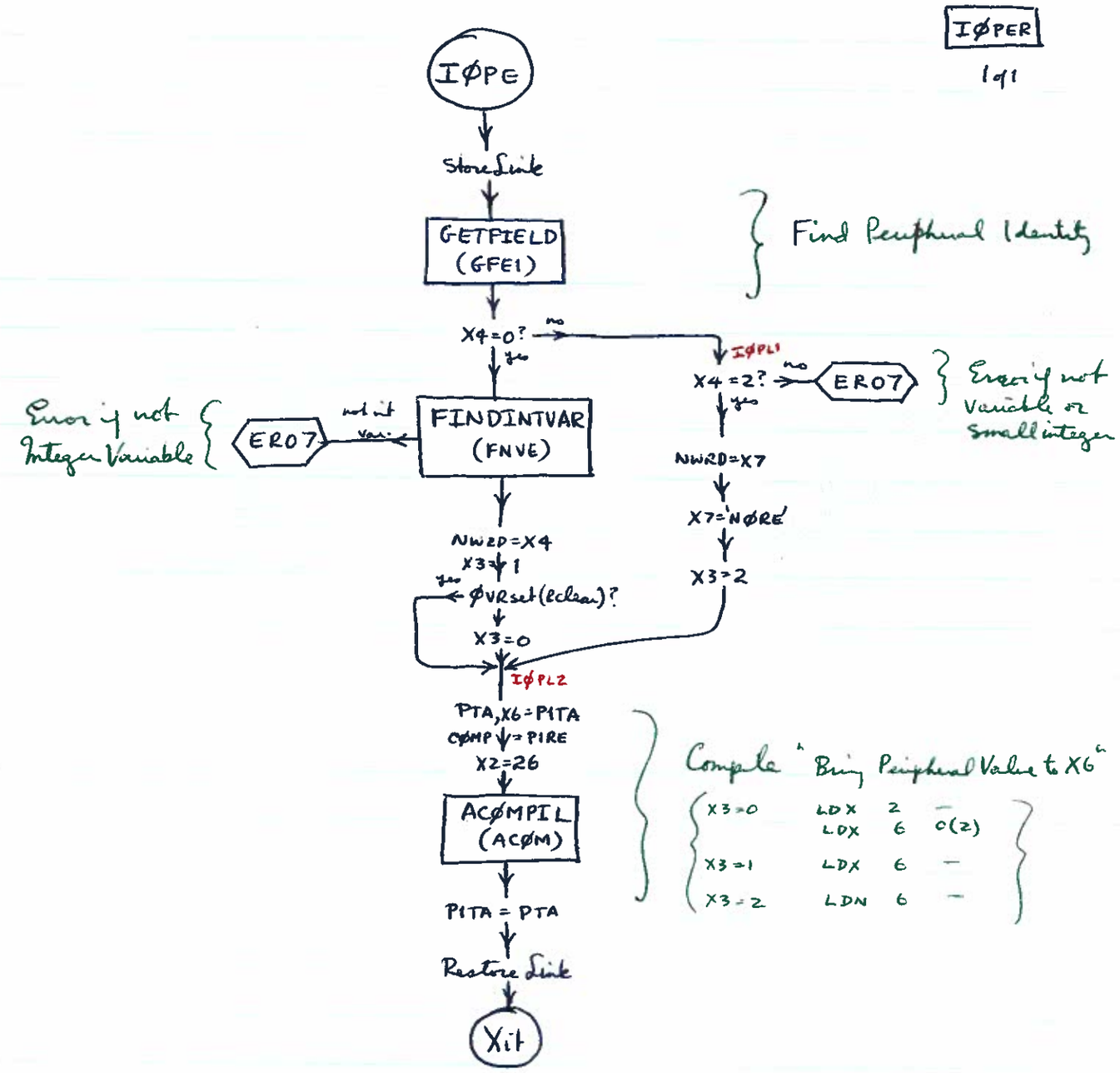


1/4/65

FJ

IØRW
1/41




IØPE
StoreLink
GETFIELD (GFE1)
Find Peripheral Identity
X4=0?
no
yes
IØPL1
X4=2?
no
ERO7
Error if not Variable or small integer
NWRD=X7
X7='NØRE'
X3=2
Error if not Integer Variable
ERO7
not int var
FINDINTVAR (FNVE)
NWRD=X4
X3=1
yes
ØVRset(Rclear)?
X3=0
IØPL2
PTA,X6=PITA
COMP=PIRE
X2=26
ACØMPIL (ACØM)
Compile "Bring Peripheral Value to X6"
X3=0 LDX 2 —
LDX 6 0(2)
X3=1 LDX 6 —
X3=2 LDN 6 —
PITA=PTA
Restore Link
Xit

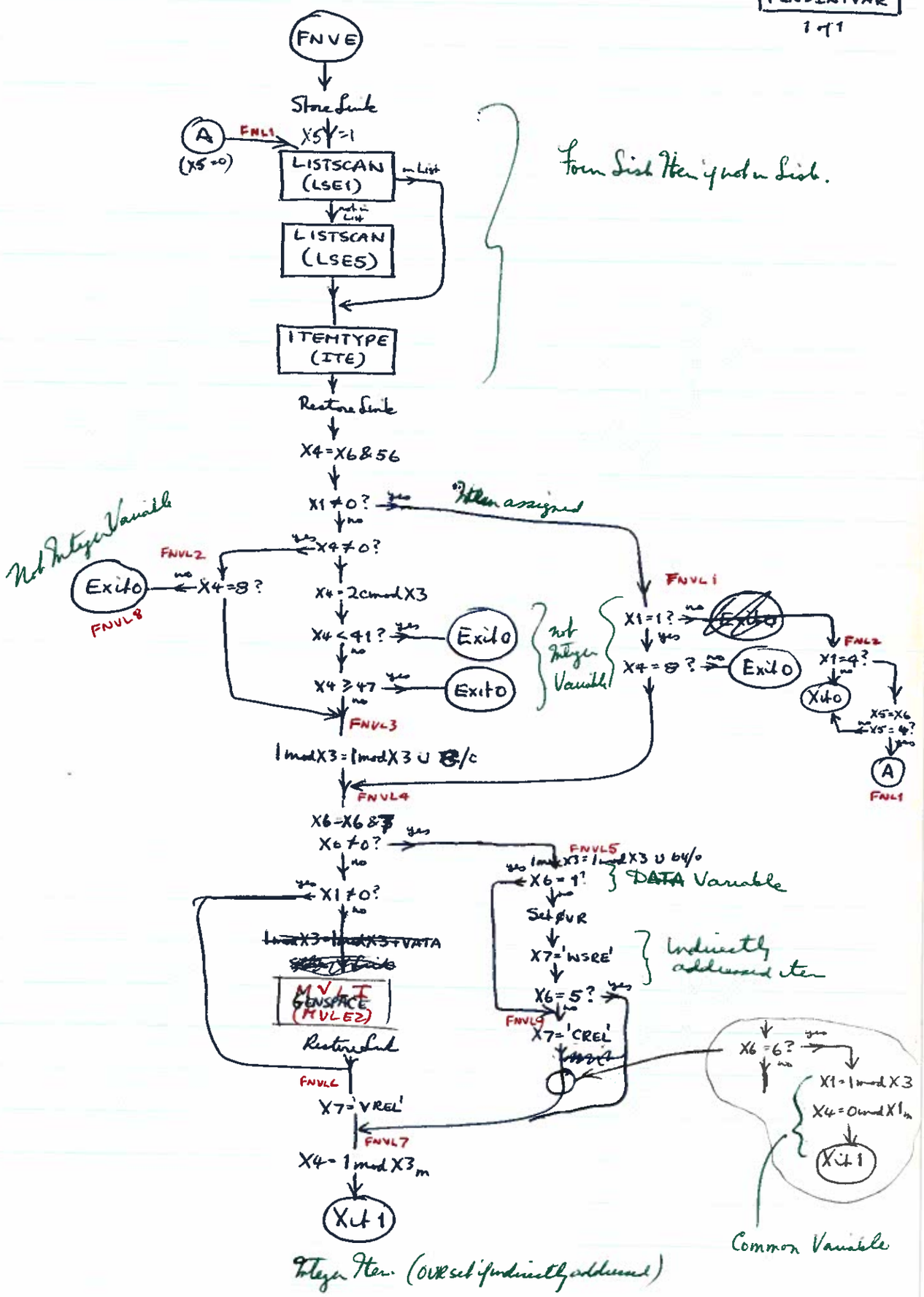
1/4/65

FORTRAN II

FH

FINDINTVAR

1 of 1

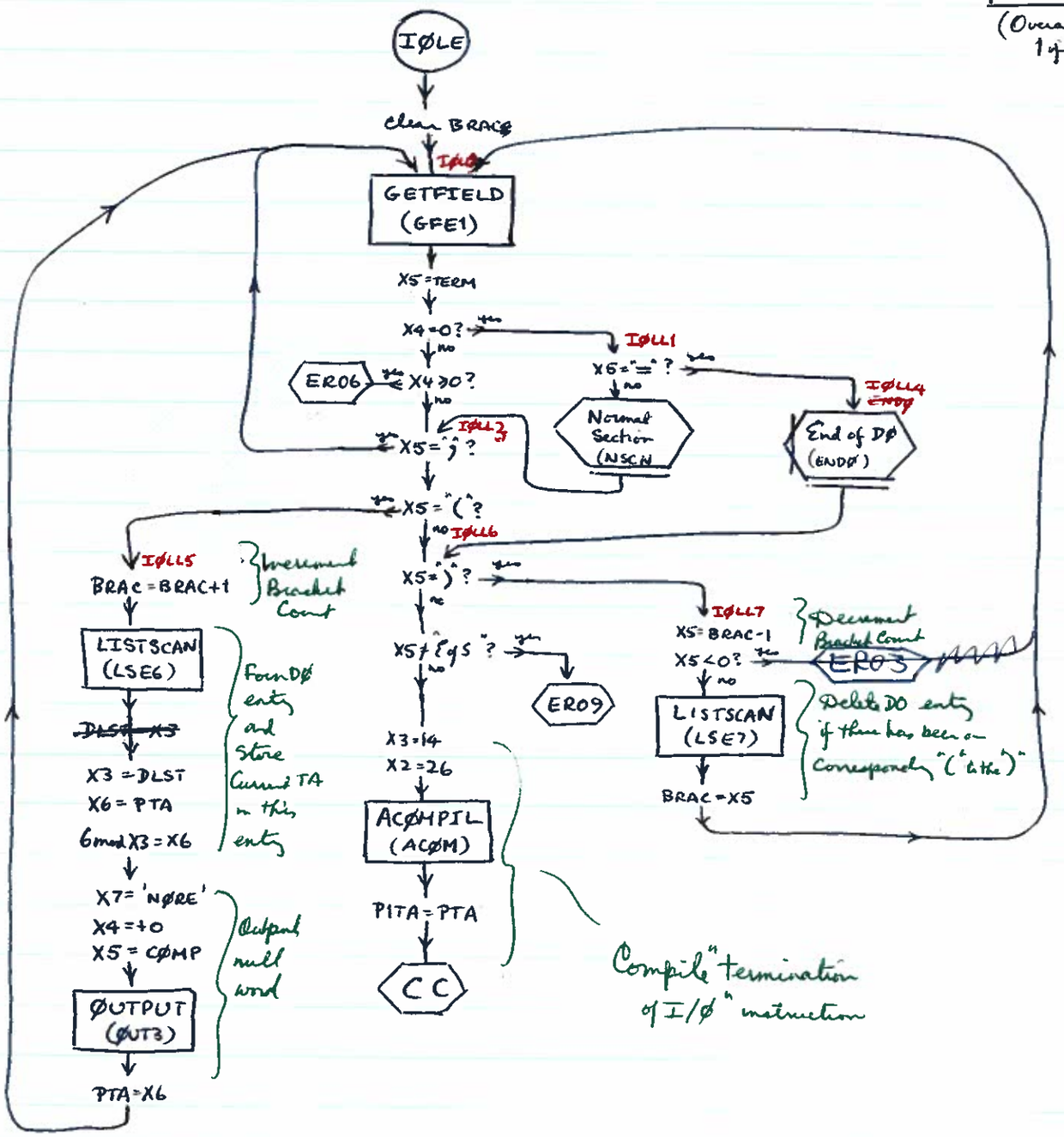


4/4/65

FØRTRAN IV

FJ

IØLIST
(Overall How)
1 of 4



Continued on Fortran ^ = in IØ list.

IØLL4

ENDØ

X3 = DLST
X6 = 6 mod X3

} X6 = Prog TA for null word

X4 = (CALL 1 P2TA)
X7 = 'P2RE'

ØUTPUT (ØUT1)

6 mod X3 = 'P1RE'/X6 ← = DØ return point

} Compile call to DØ prologue (to be compiled in Prog. Blk 2.

P1TA = PTA
PTA, X6 = P2TA
COMP = X7

DØ 1 (DØEE)

X6 = PTA
X4 = (EXIT 1 0)
X5 = 'P2RE'
X7 = 'NØRE'

ØUTPUT (ØUT3)

} Compile Prologue to DØ in Program Block 2 & end with EXIT instruction

P2TA = X6
~~PTA~~
X1 = DLST
Set ØVR

DØ 2 (ØØ22)

COMP = 'P1RE'
X5 = TERM - 25
PTA = P1TA

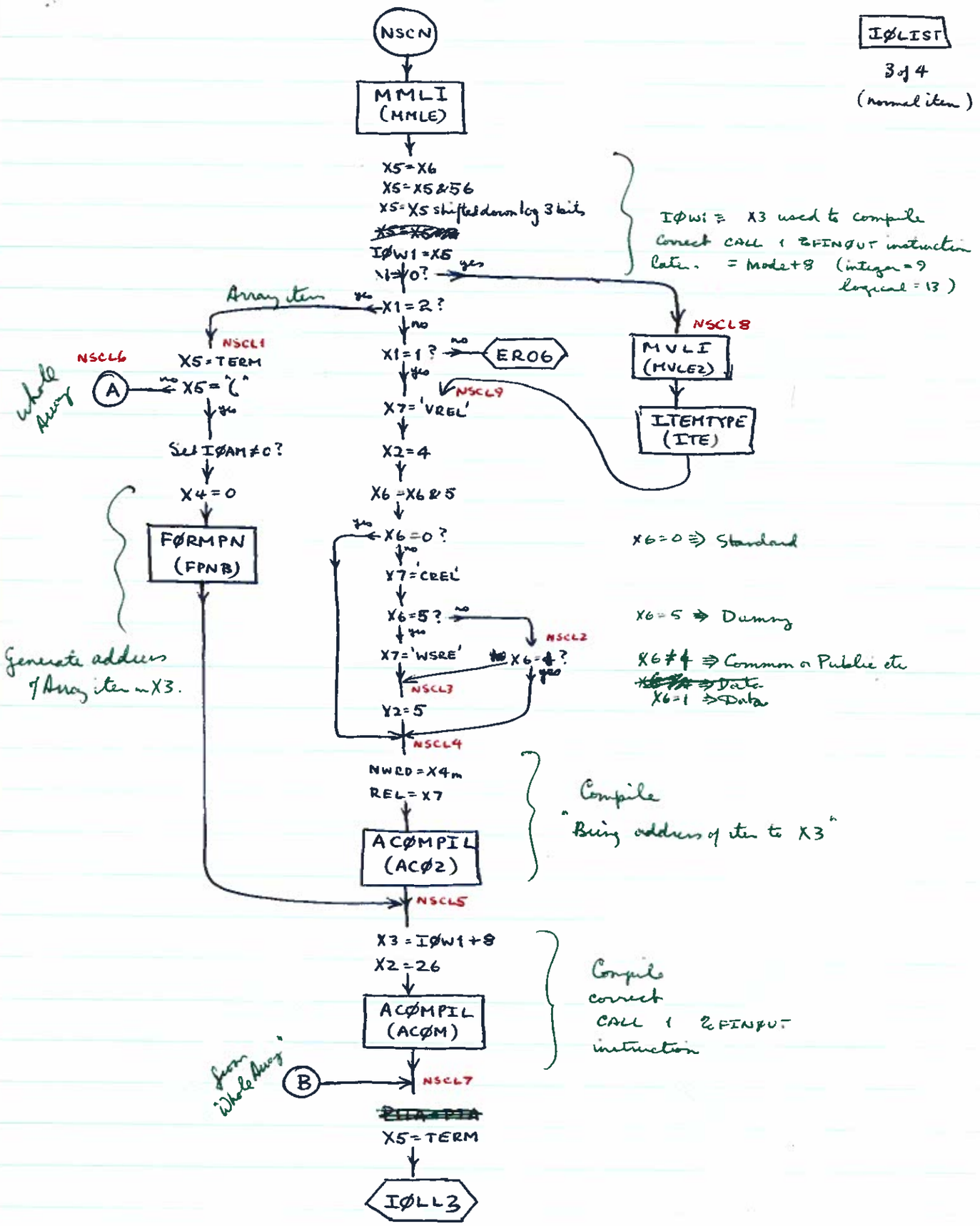
IØLL6

} Output DØ epilogue in Prog Block 1. (OVR cleared in DØ 2)

Return to look for ")" etc.

4/4/65

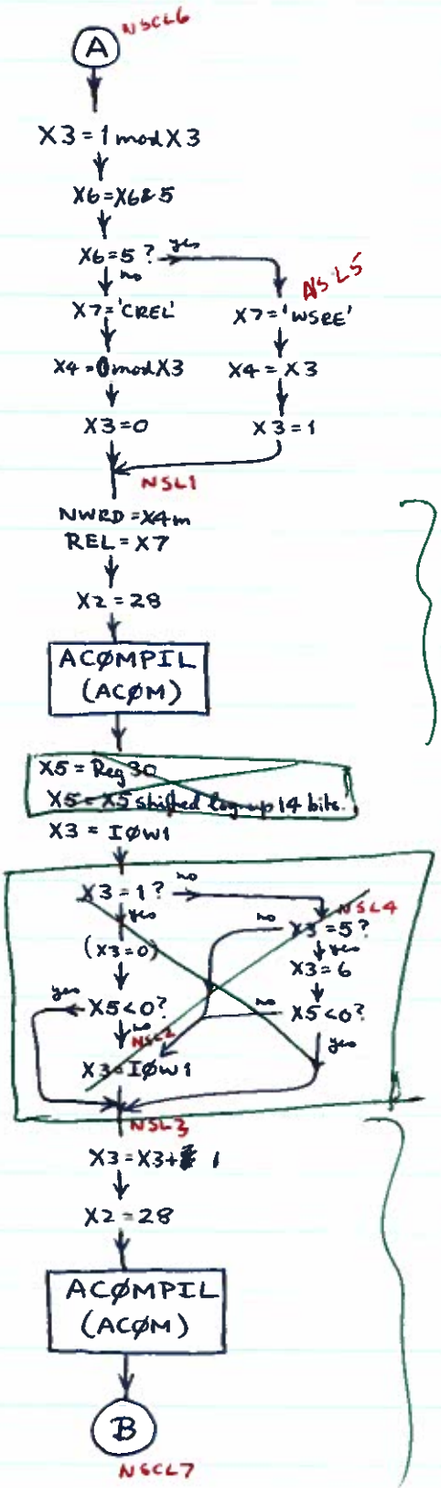
IØLIST
3 of 4
(normal item)



4/4/65

FØRTRAN IV

IØLIST
4 of 4
(whole Array)



Output instruction to put address of ~~Array into X3~~ Array Header into X3

Compile CALL 1 2 FIØTA+n+2n

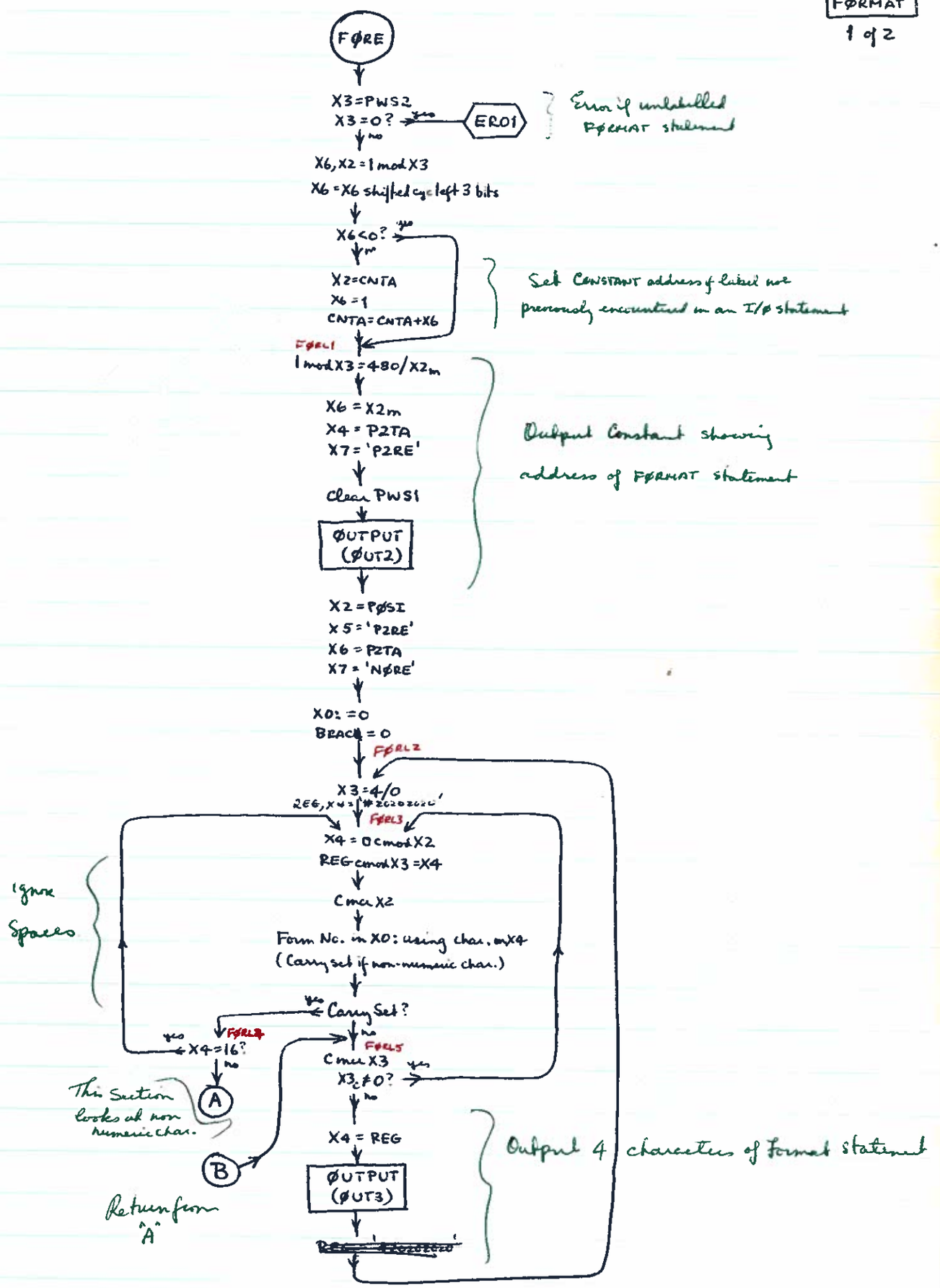
n=0 ~~Word integer~~ Integer
1 ~~Standard integer~~ Real
~~2 Real~~ DP
3 ~~D.P~~ Cplx
~~4 Complex~~ 4 Log.
~~5 Logical (Standard)~~
6 1 word logical

4/4/66

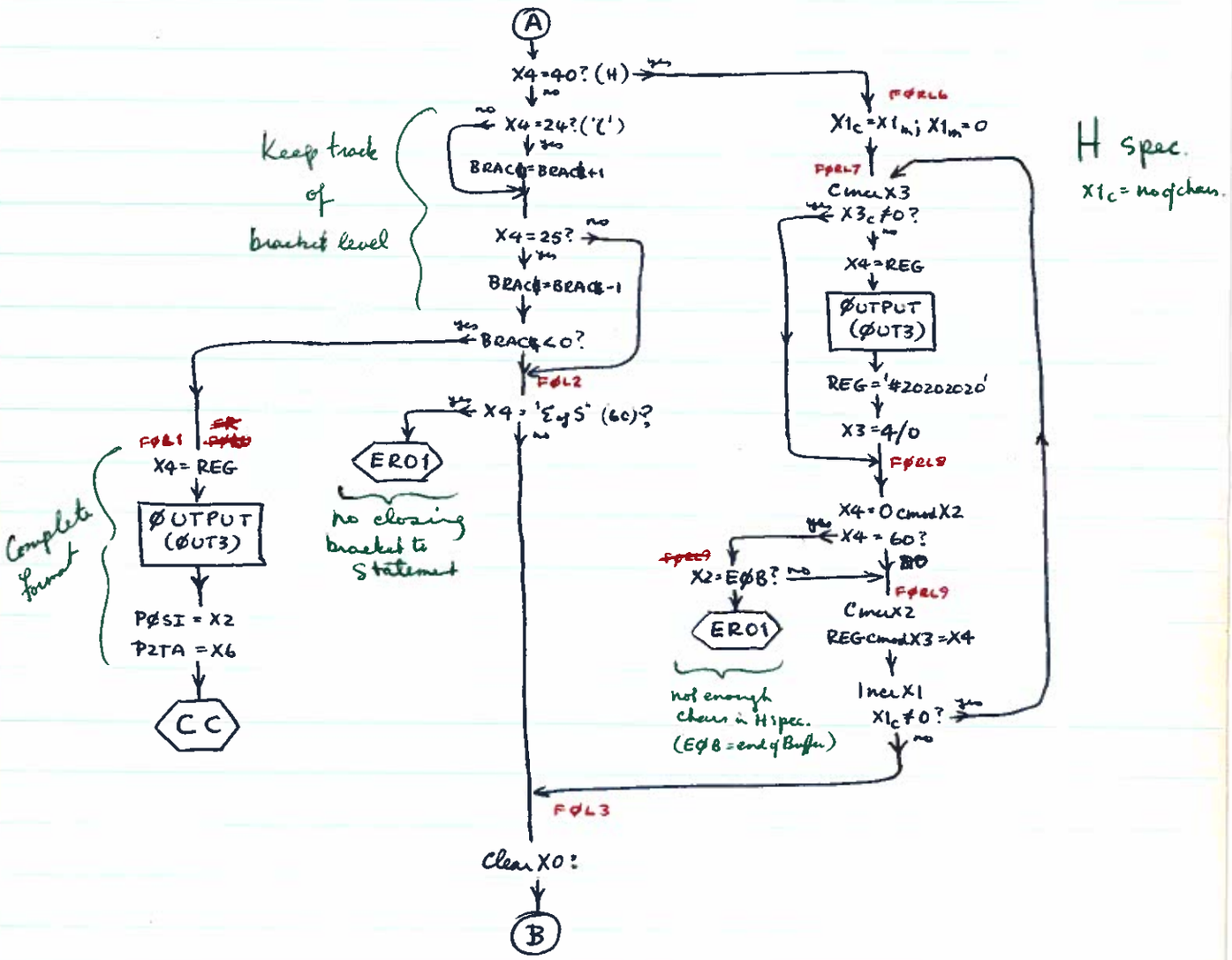
FØRTRAN IV

(FJ)

FØRMAT
1 of 2



FØRTRAN IV


A
X4=40? (H)
yes
no
X4=24? ('(')
yes
BRACE=BRACE+1
Keep track of bracket level
X4=25?
yes
BRACE=BRACE-1
BRACE<0?
yes
FØL2
X4='ΣgS' (60)?
yes
ERO1
no closing bracket to Statement
FØL1
X4=REG
ØUTPUT (ØUT3)
Complete format
PØSI=X2
P2TA=X6
CC
no
FØRL6
X1c=X1m; X1m=0
FØRL7
CmcuX3
X3c≠0?
yes
no
X4=REG
ØUTPUT (ØUT3)
REG='#20202020'
X3=4/0
FØRL8
X4=0 cmcuX2
X4=60?
yes
X2=EØB?
no
BO
FØRL9
CmcuX2
REG-cmcuX3=X4
ERO1
not enough chars in H spec. (EØB=end of Buffer)
IncuX1
X1c≠0?
yes
no
FØL3
Clear X0:
B
H Spec.
X1c=no of chars.

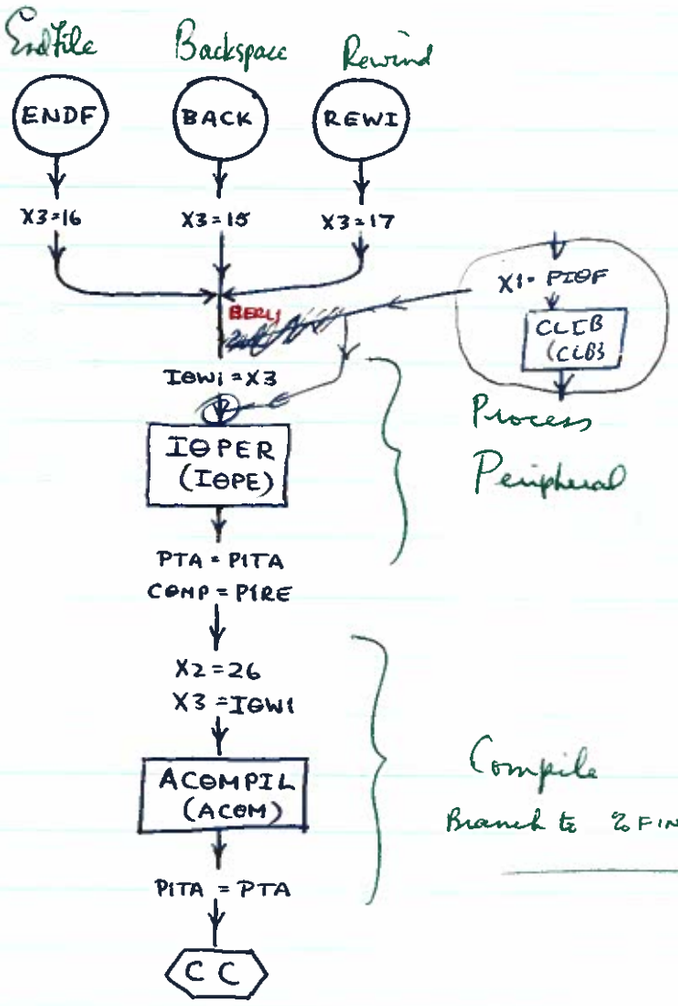
6/4/65

FORTRAN IV

FJ

IOMISC

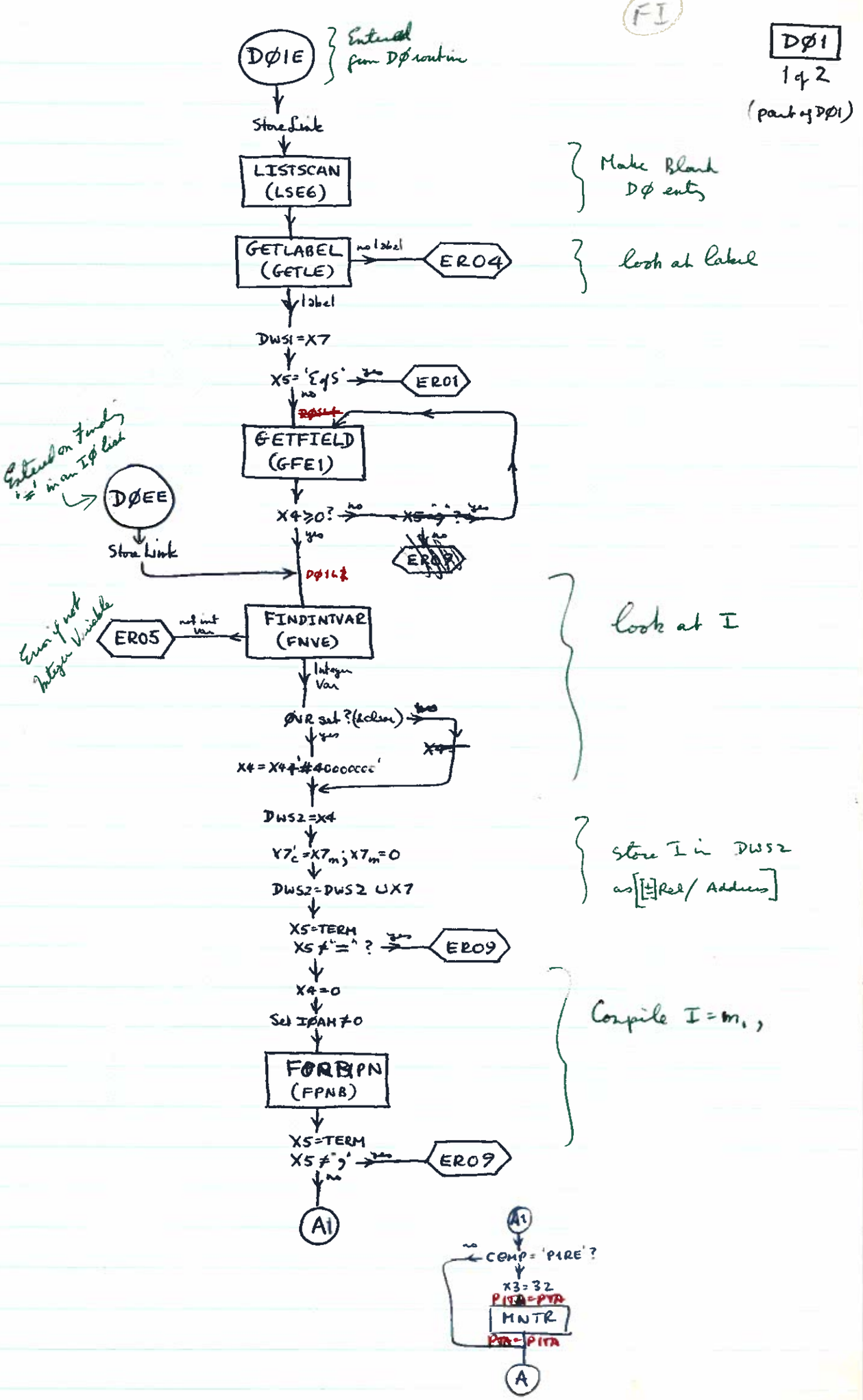
111

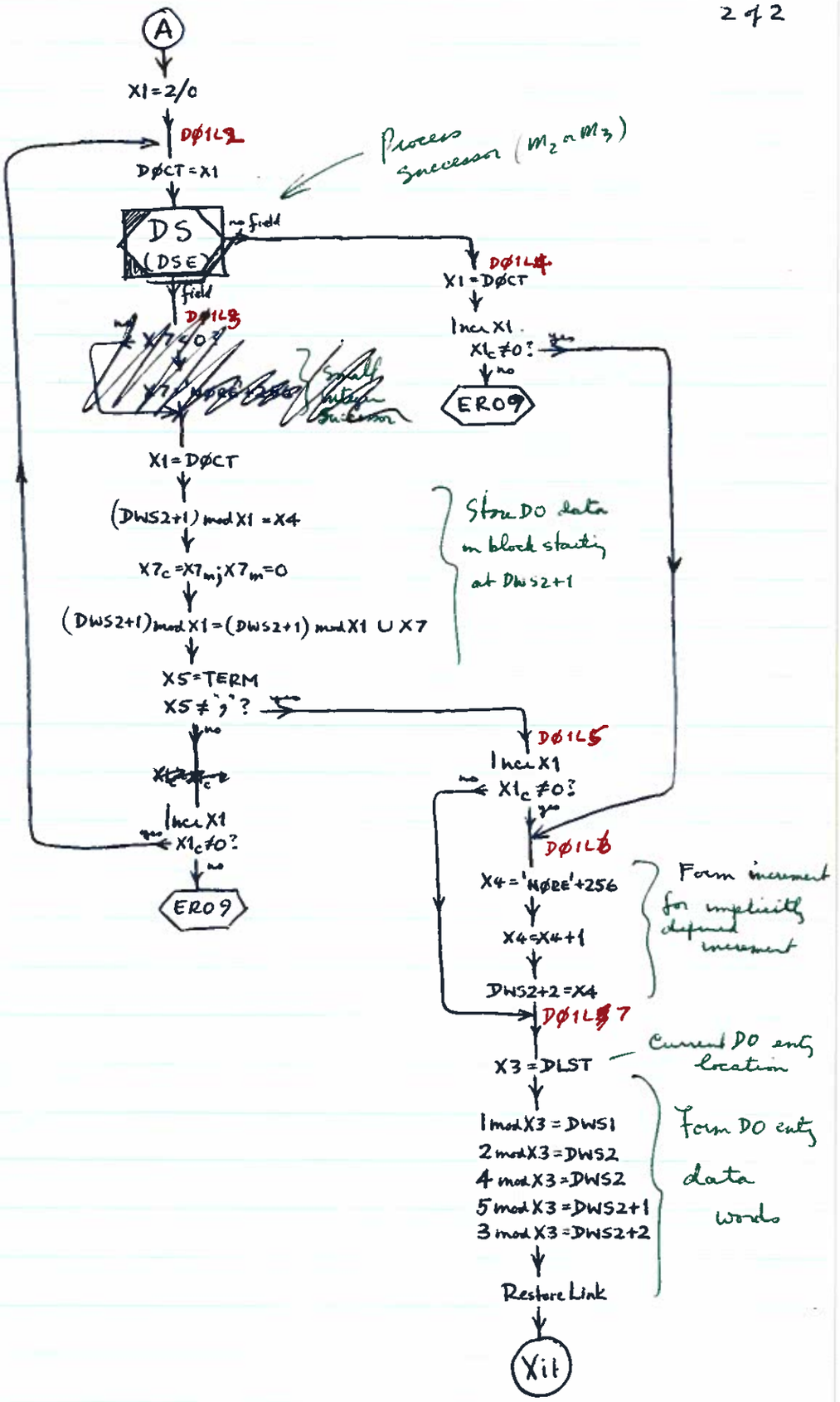


FORTRAN IV

FI

DØ1
1 of 2
(part of DØ1)




A
X1=2/0
DØ1L2
DØCT=X1
Process successor (M2 or M3)
DS
(DSE)
no field
field
DØ1L4
X1=DØCT
Incr X1
X1c≠0?
yes
no
ERO9
DØ1L3
X7≠0?
X1=DØCT
(DWS2+1) mod X1 =X4
X7c=X7m; X7m=0
(DWS2+1) mod X1=(DWS2+1) mod X1 ∪ X7
Store DO data in block starting at DWS2+1
X5=TERM
X5≠',' ?
DØ1L5
Incr X1
X1c≠0?
DØ1L6
X4='NØRE'+256
X4=X4+1
DWS2+2=X4
Form increment for implicitly defined increment
DØ1L7
X3=DLST
Current DO entry location
1 mod X3 = DWS1
2 mod X3 = DWS2
4 mod X3 = DWS2
5 mod X3 = DWS2+1
3 mod X3 = DWS2+2
Form DO entry data words
Restore Link
Xit

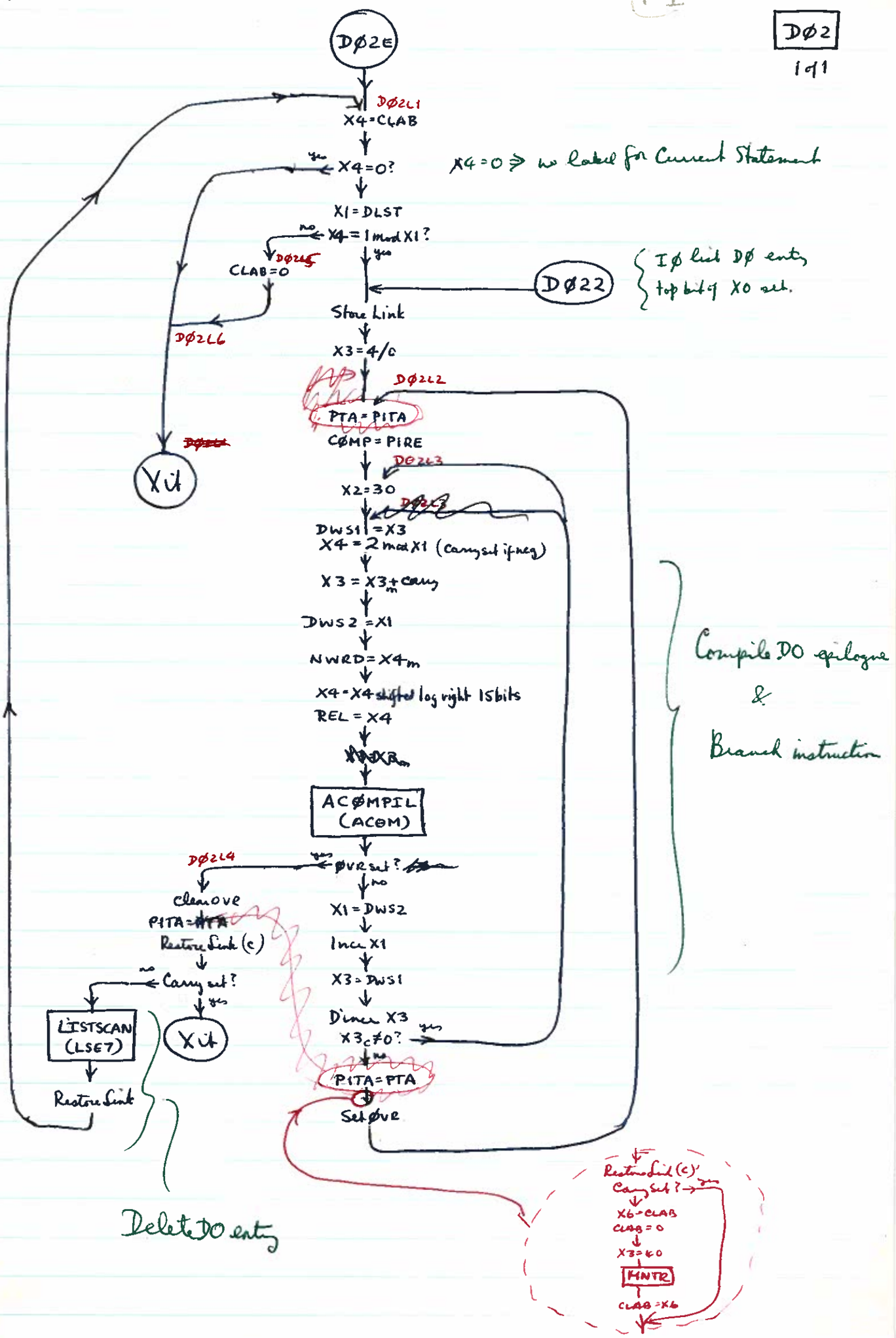
6/4/65

FORTRAN IV

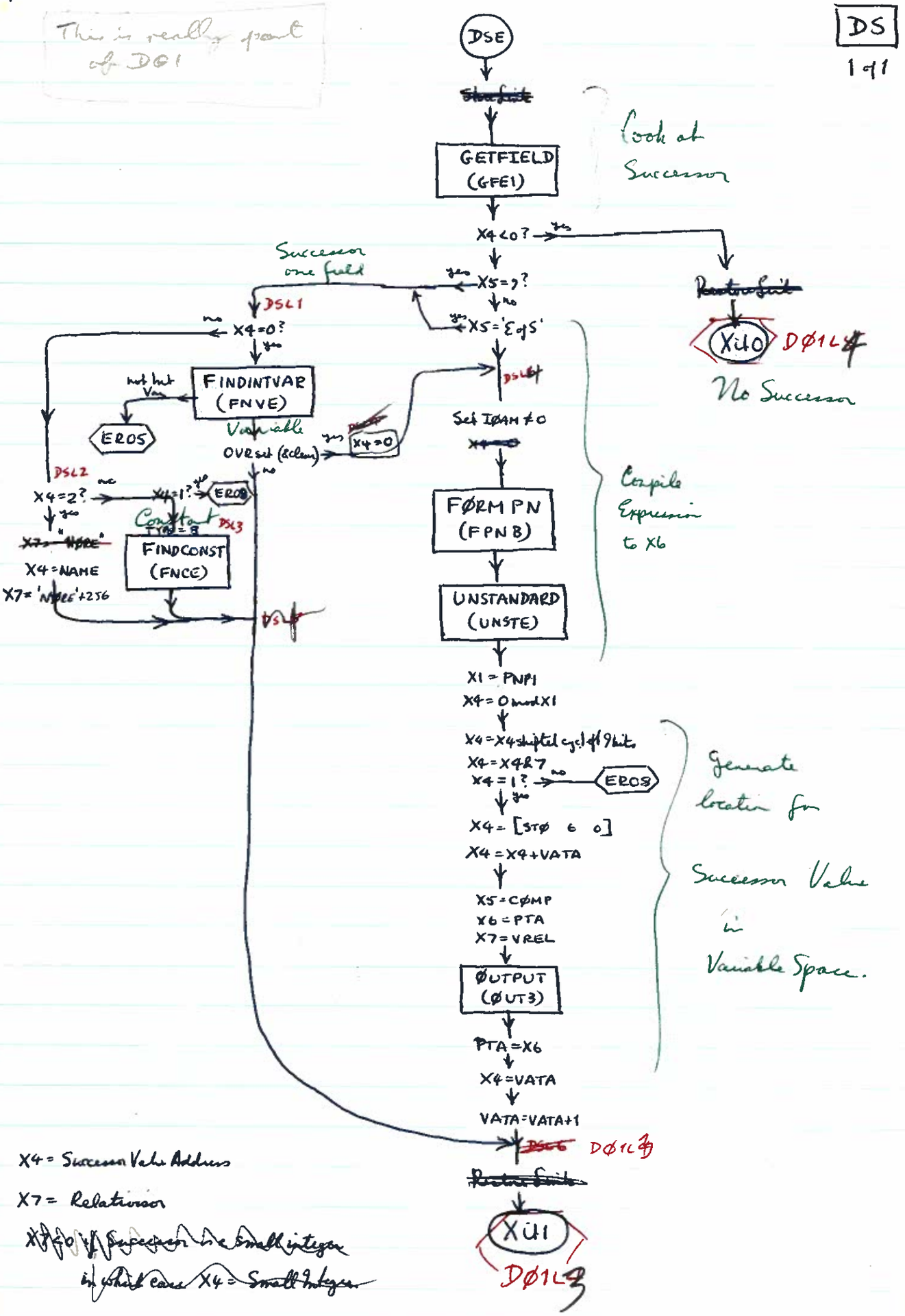
FI

DØ2

191



This is really part of DO1



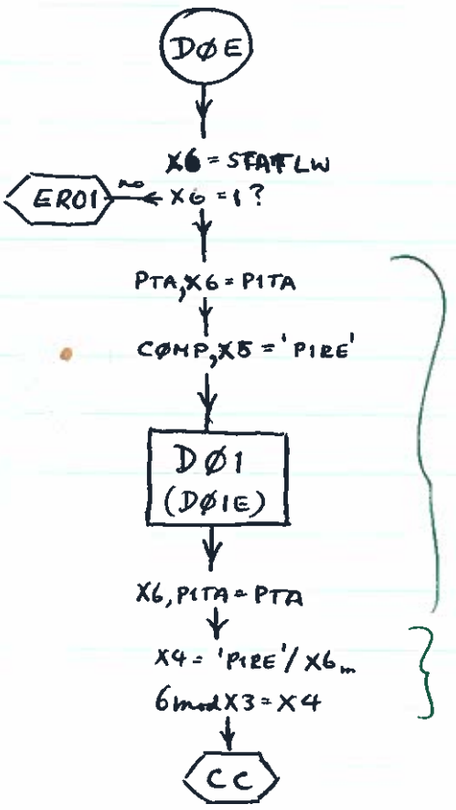
7/4/65

FØRTRAN IV

FI

DØ

141



DØ not allowed as successor statement in Logical IF.

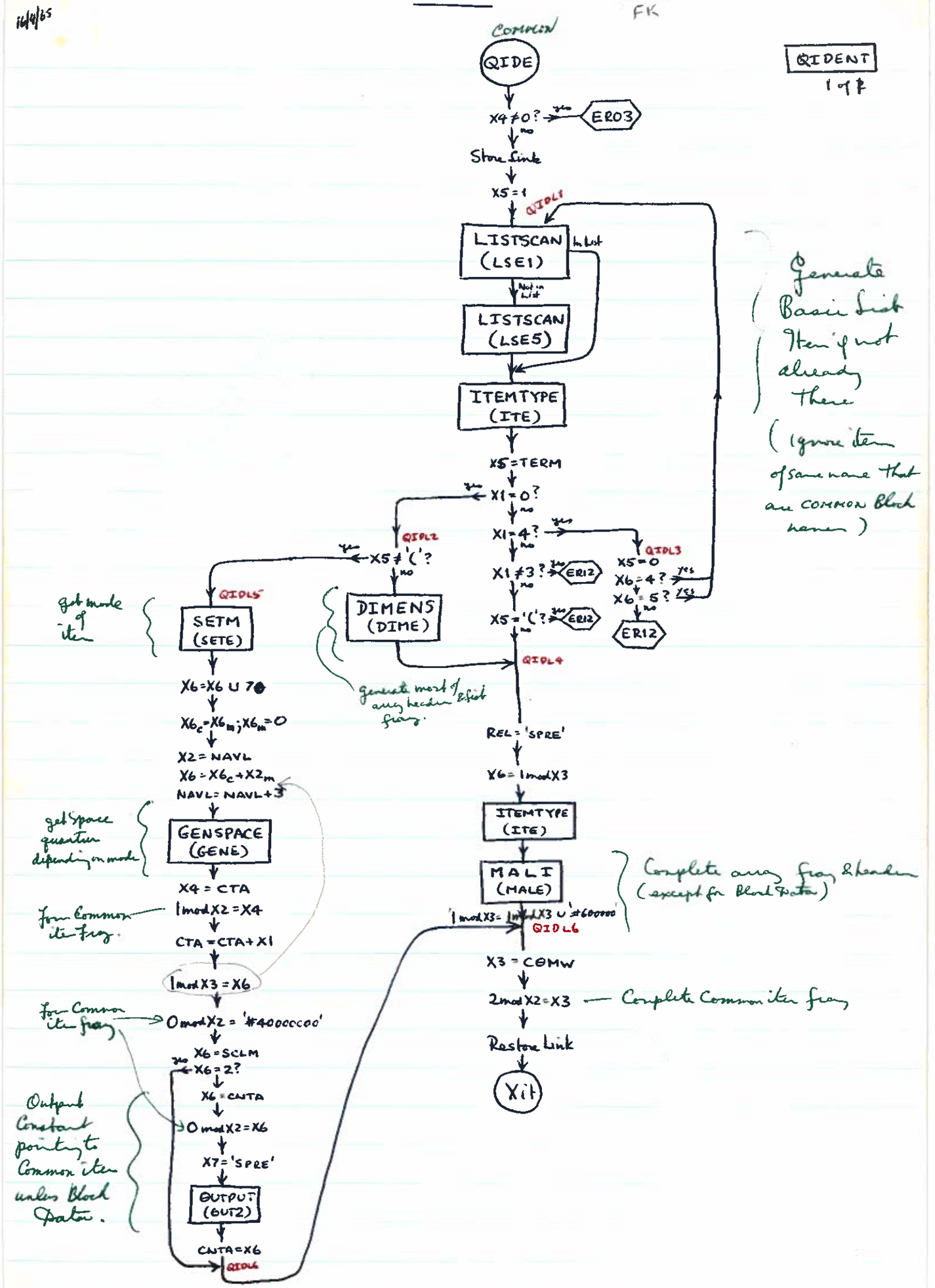
Generate DØ list item and compile Prologue in Prog Blk 1 space

Generate Return address item in DØ list

16/9/65

FORTRAN IV

FK

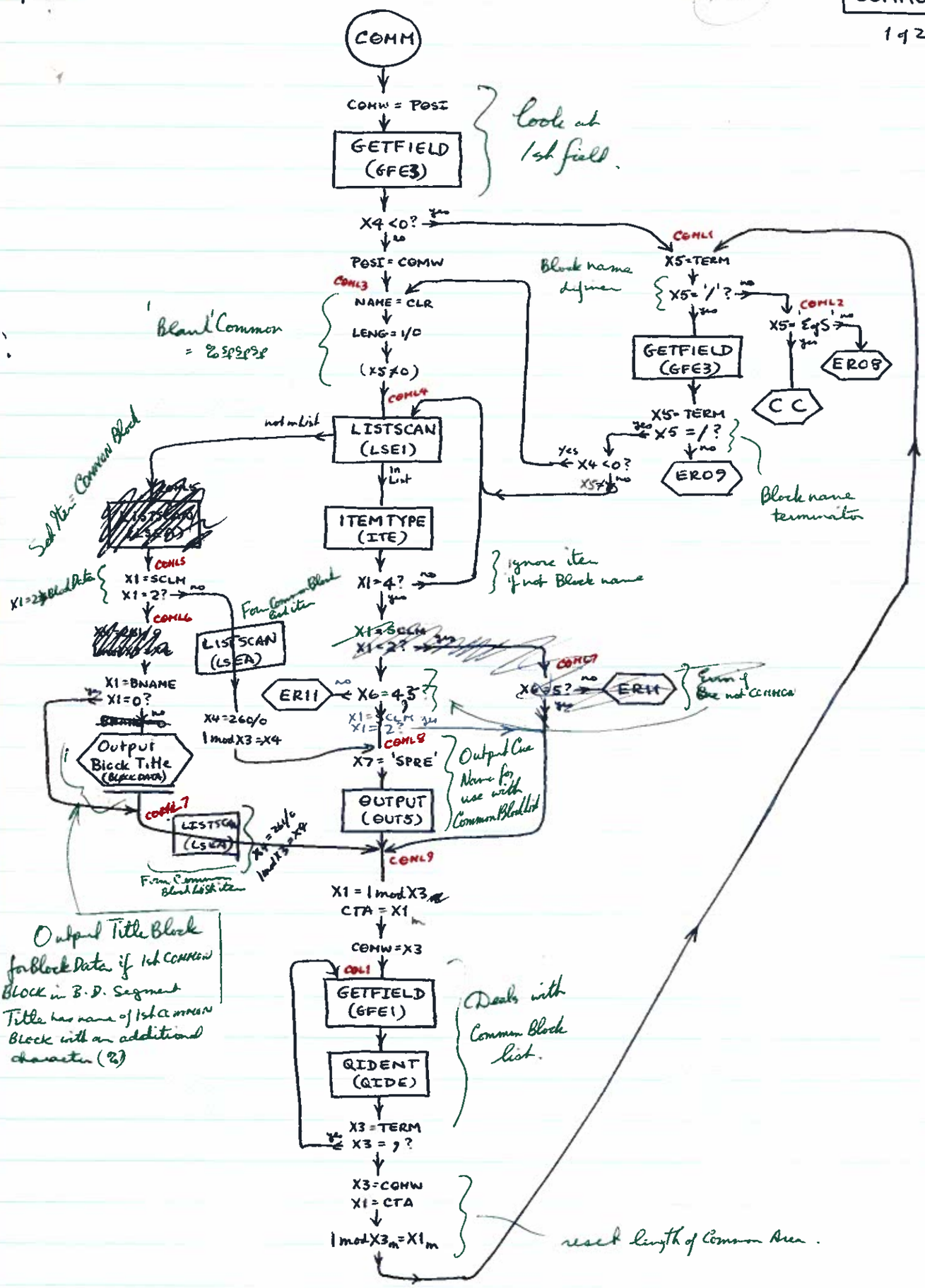


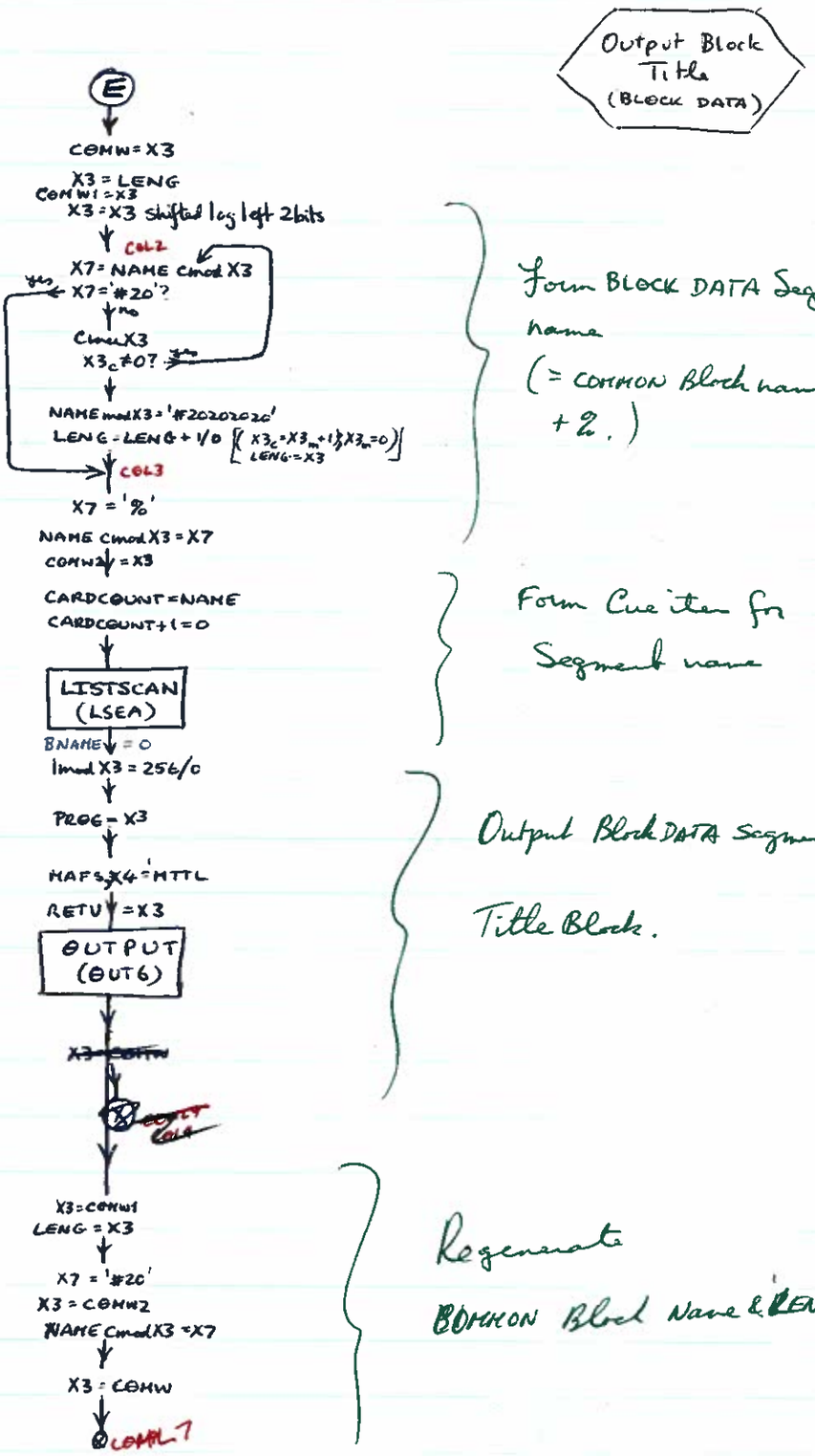
16/8/65

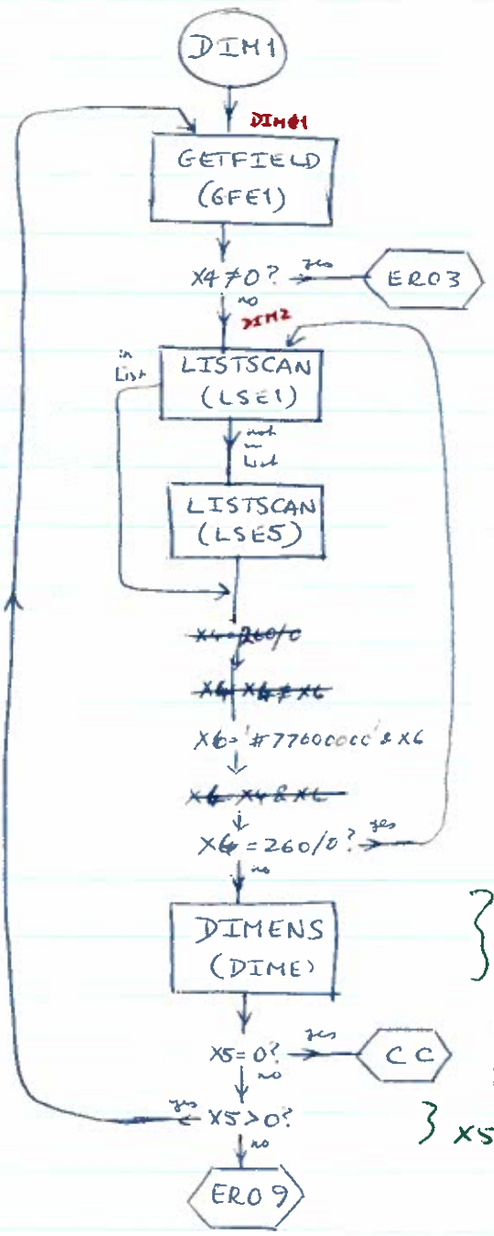
FK

COMMON

1 of 2




Output Block Title (BLOCK DATA)
COMMW=X3
X3=LENG
COMW1=X3
X3=X3 shifted log left 2bits
COL2
X7=NAME CmodX3
X7='#20'?
yes
no
CmodX3
X3c≠0?
yes
NAME modX3='#20202020'
LENG=LENG+1/0 [(X3c=X3m+1 ; X3m=0) LENG=X3]
COL3
X7='%'
NAME CmodX3=X7
COMW2=X3
CARDCOUNT=NAME
CARDCOUNT+1=0
LISTSCAN (LSEA)
BNAME=0
ImodX3=256/0
PROG=X3
MAFS,X4='HTTL
RETU=X3
OUTPUT (OUT6)
X3=COMW
COL7
X3=COMW1
LENG=X3
X7='#20'
X3=COMW2
NAME CmodX3=X7
X3=COMW
COL7
Form BLOCK DATA Segment name (= COMMON Block name + 2.)
Form Cue item for Segment name
Output Block DATA Segment Title Block.
Regenerate BOMMON Block Name & 'LENG'



Array name

Form List item if not already formed (X3= List address)

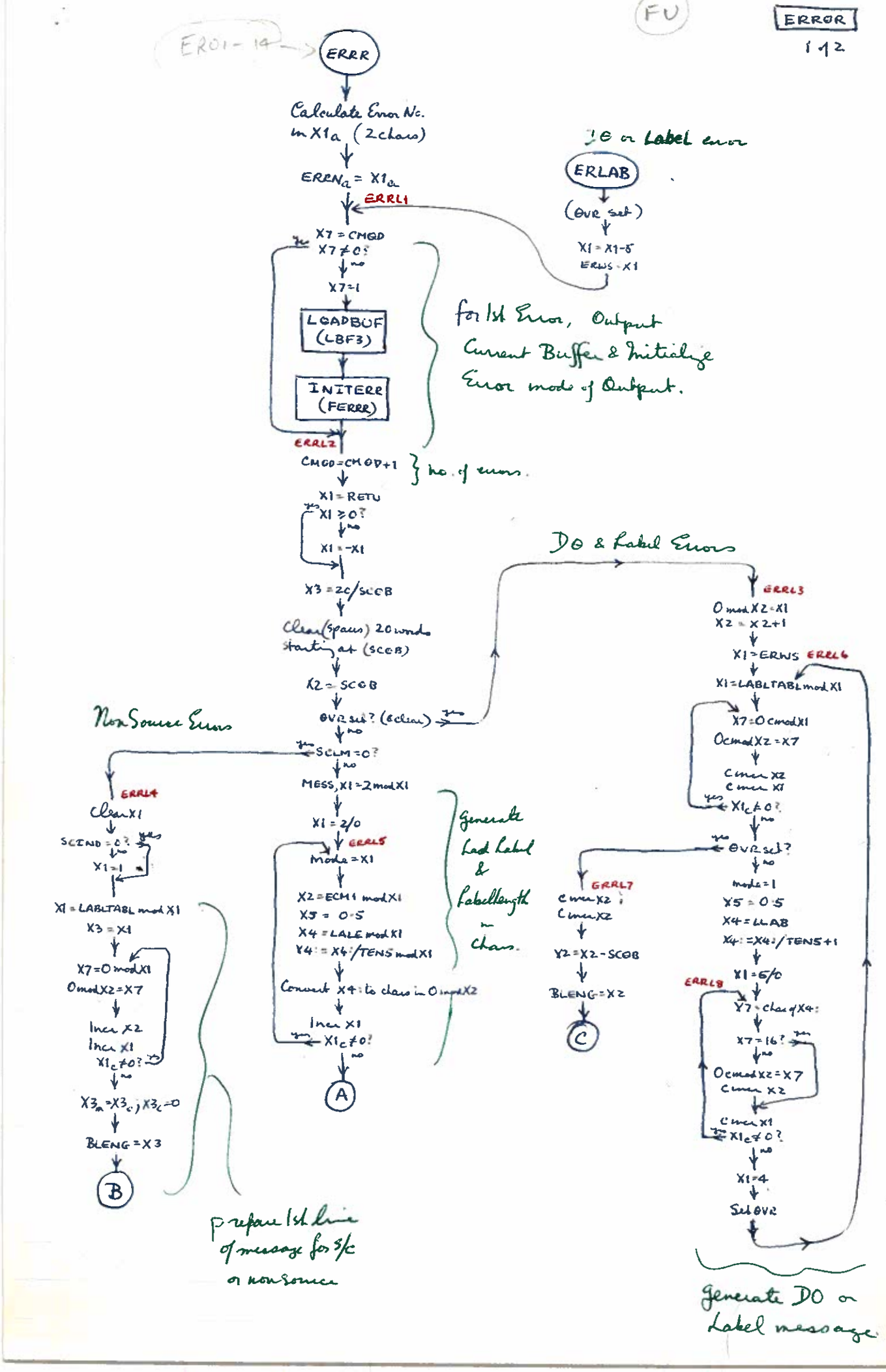
Ignore COMMON Block name

Deals with Dimensioning

End of Statement

X5=0

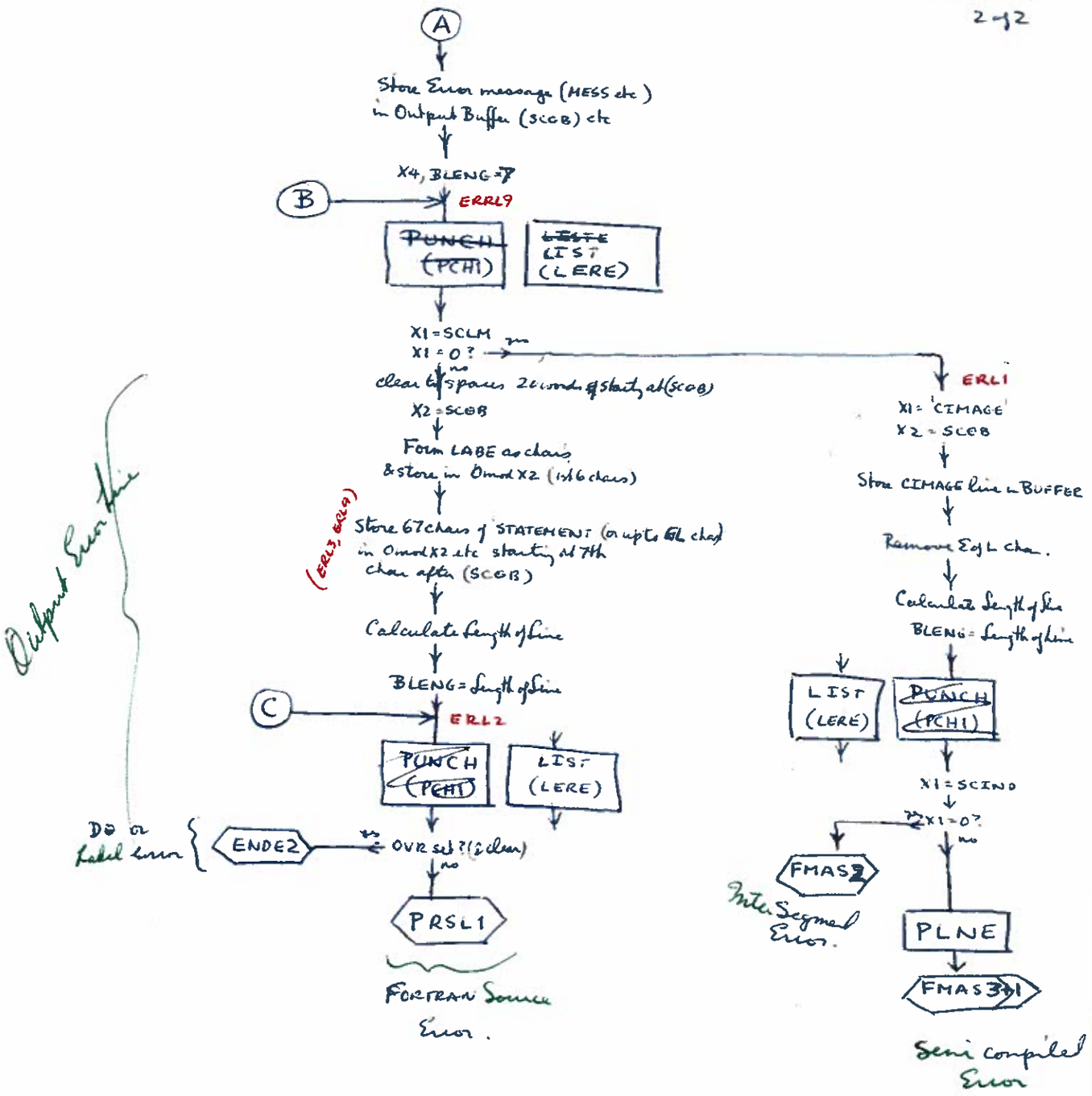
X5>0 ⇒ ',' as separator


ERO1 - 14
ERR
Calculate Error No. in X1a (2 chars)
ERRNa = X1a
ERRL1
X7 = CMOD
X7 ≠ 0?
X7=1
LOADBUF (LBF3)
INITERR (FERR)
ERRL2
CMOD=CMOD+1
X1=RETU
X1 ≥ 0?
X1=-X1
X3=2c/SCOB
Clear (spaces) 20 words starting at (SCOB)
X2=SCOB
OVR set? (&clear)
SCLM=0?
MESS, X1=2 mod X1
X1=2/0
ERRL5
Mode=X1
X2=ECM1 mod X1
X3=0·5
X4=LALE mod X1
X4=X4/TENS mod X1
Convert X4 to chars in O mod X2
Incr X1
X1c≠0?
A
Generate Last Label & Label length in chars.
ERRL4
Clear X1
SCIND=0?
X1=1
X1=LABLTABL mod X1
X3=X1
X7=O mod X1
O mod X2=X7
Incr X2
Incr X1
X1c≠0?
X3m=X3c) X3c=0
BLENG=X3
B
prepare 1st line of message for s/c or non source
Non Source Errors
DO & Label Errors
ERRL3
O mod X2=X1
X2=X2+1
X1=ERNS
ERRL6
X1=LABLTABL mod X1
X7=O cmod X1
O cmod X2=X7
C mem X2
C mem X1
X1c≠0?
OVR set?
ERRL7
C mem X2
C mem X2
Y2=X2-SCOB
BLENG=X2
C
mode=1
X5=0·5
X4=LLAB
X4=X4/TENS+1
X1=6/0
ERRL8
Y7=char of X4
X7=16?
O cmod X2=X7
C mem X2
C mem X1
X1c≠0?
X1=4
Set OVR
Generate DO or Label message
ERLAB
(OVR set)
X1=X1-5
ERNS=X1
DO or Label error

26/4/65.

FORTRAN IV

ERROR
2 of 2

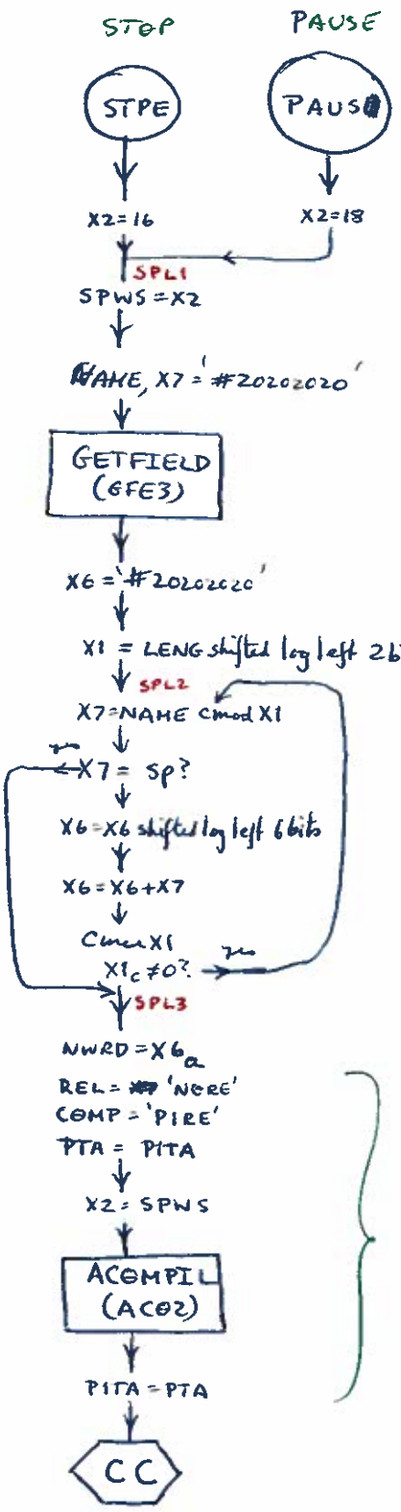


27/4/67

FORTRAN IV

FF

PAUSE
STOP



Look for field.

Find last 2 chars following PAUSE or STOP
(eg PAUSE 1234 finds 34)

Output PAUSE or STOP instruction

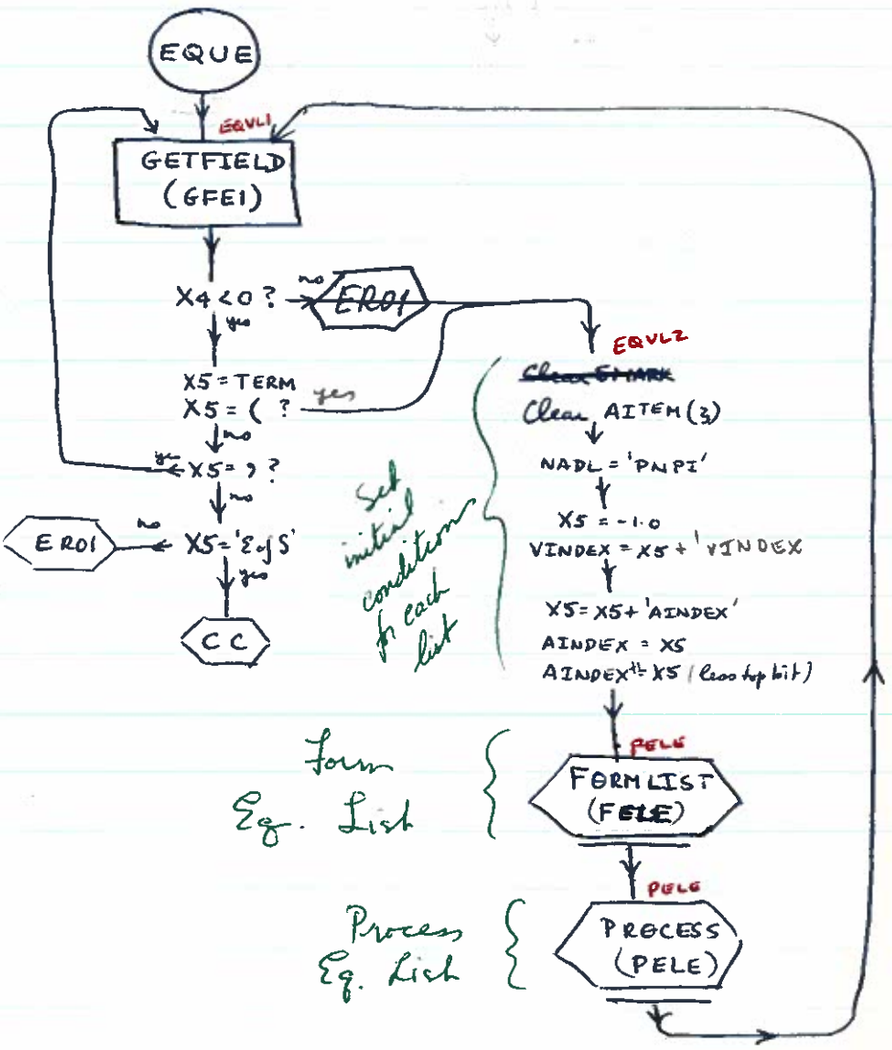
3/5/65

FORTRAN IV

FG

EQUIVALENCE

(General Flow)



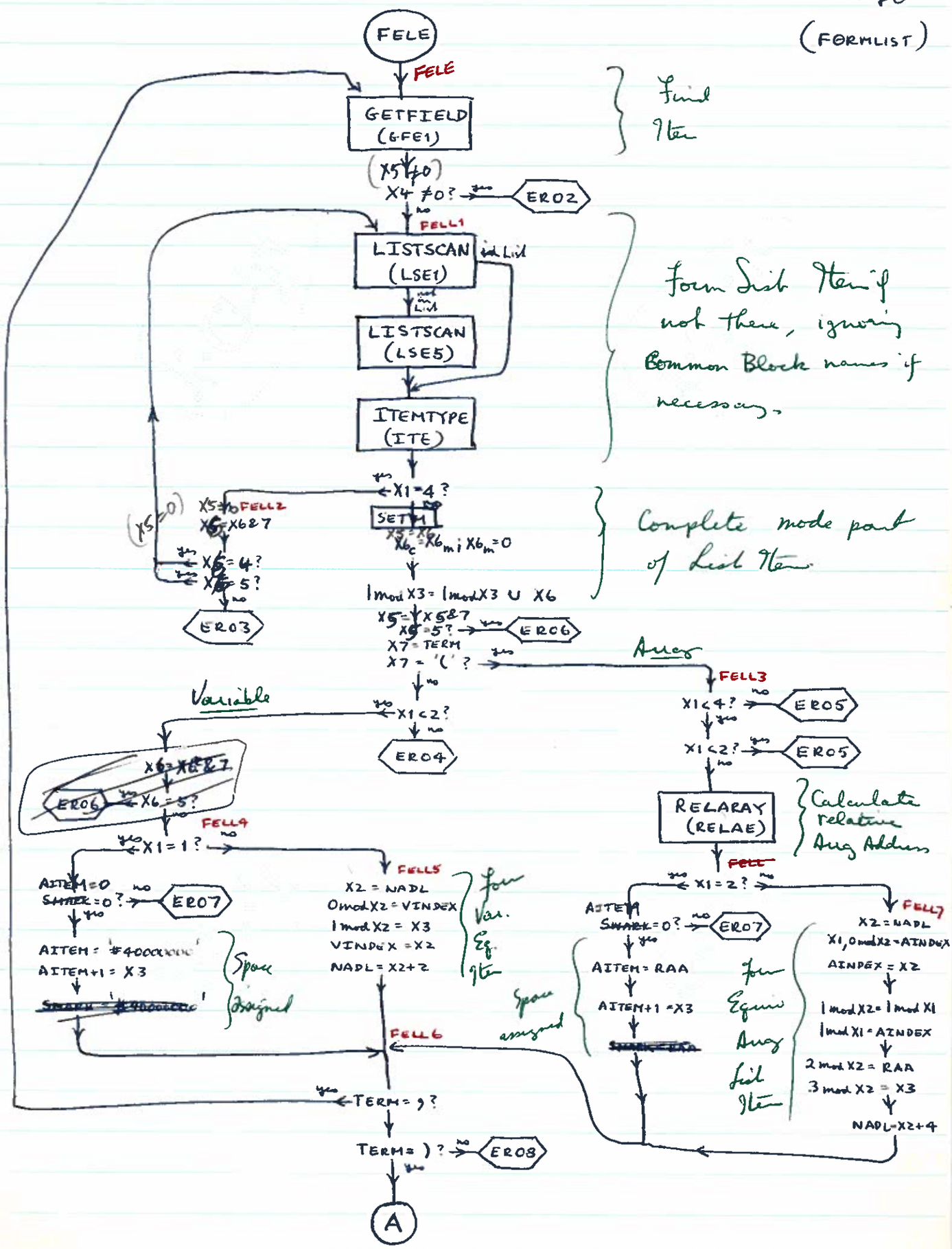
3/5/65

FORTRAN IV

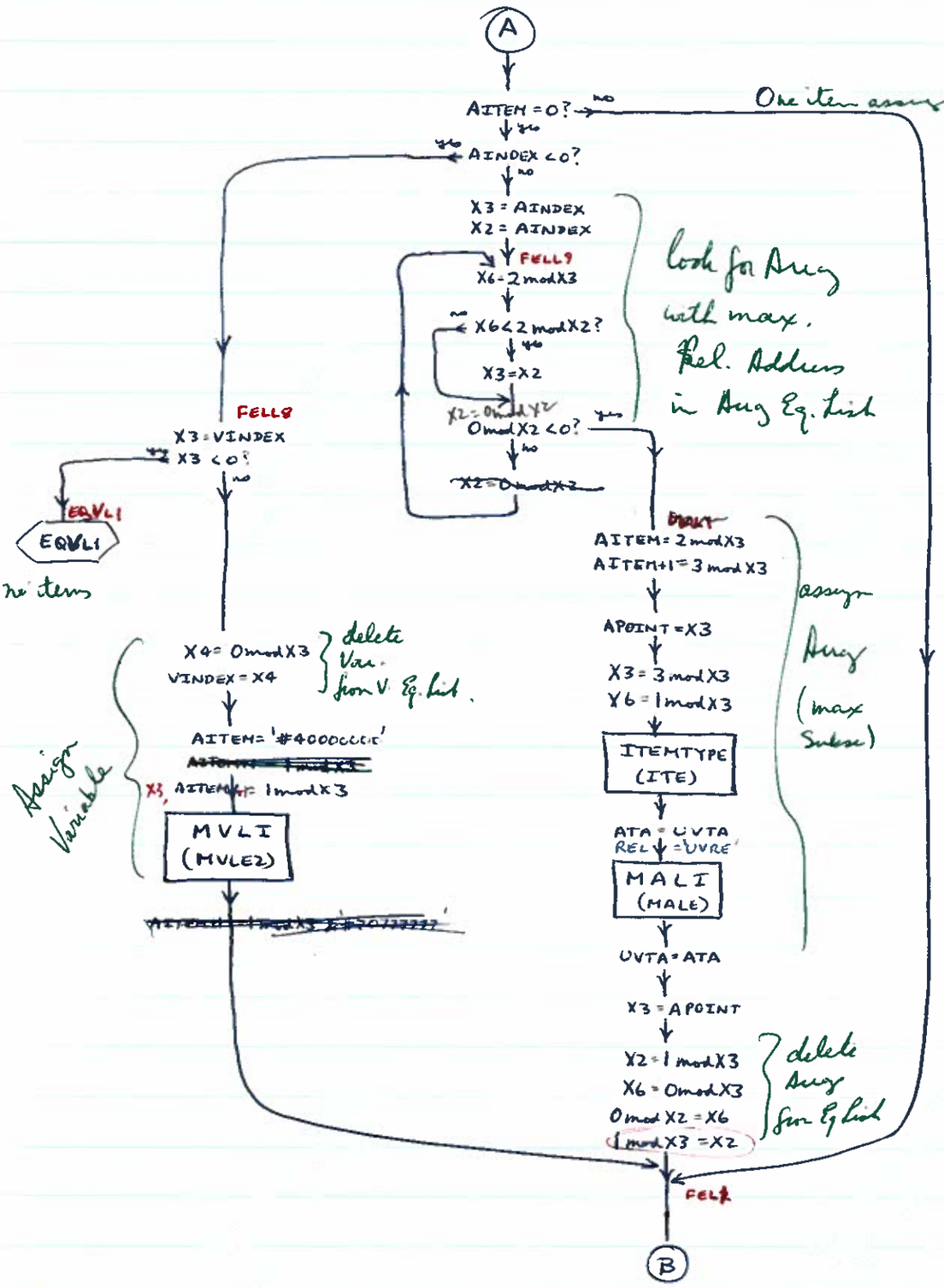
EQUIVALENCE

246

(FORMLIST)



Put list in Standard Form.

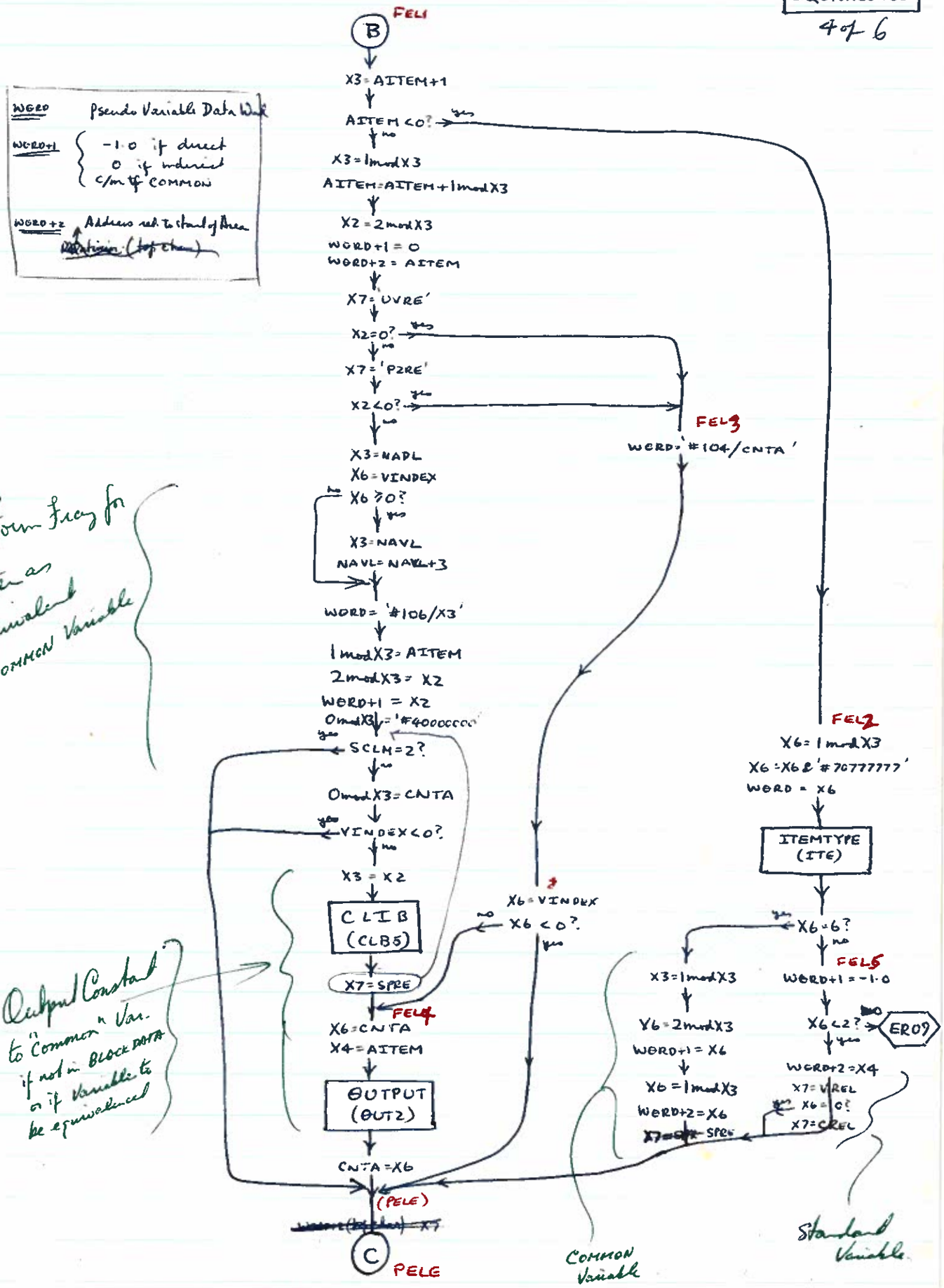

A
AITEM = 0?
no
One item assigned
yes
AINDEX < 0?
yes
no
X3 = AINDEX
X2 = AINDEX
FELL9
X6 = 2 mod X3
X6 < 2 mod X2?
yes
X3 = X2
X2 = 0 mod X2
0 mod X2 < 0?
yes
no
look for Array with max. Rel. Address in Array Eq. List
FELL9
X3 = VINDEX
X3 < 0?
yes
no
EQVL1
EQVL1
no items
FELL1
AITEM = 2 mod X3
AITEM+1 = 3 mod X3
APOINT = X3
X3 = 3 mod X3
Y6 = 1 mod X3
ITEMTYPE (ITE)
ATA = UVTA
REL = UVRE
MALI (MALE)
UVTA = ATA
X3 = APOINT
assign Array (max Subscr)
X4 = 0 mod X3
VINDEX = X4
delete Var. from V. Eq. List.
AITEM = '#4000CCCC'
X3, AITEM+1 = 1 mod X3
Assign Variable
MVLI (MVLE2)
Y2 = 1 mod X3
X6 = 0 mod X3
0 mod X2 = X6
1 mod X3 = X2
delete Array from Eq List
FELL
B
Put Assigned item in Standard form.

3/5/65

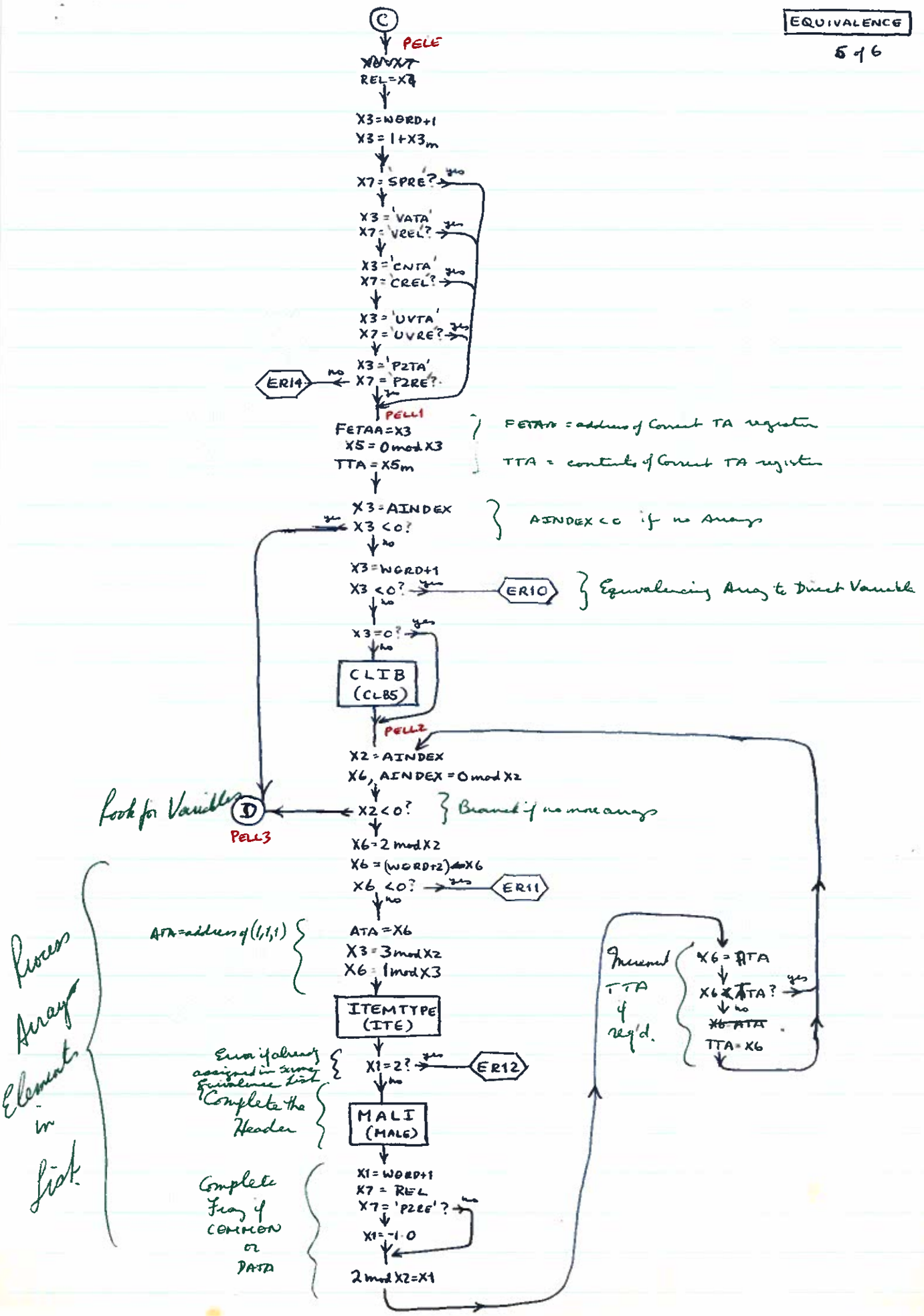
FORTRAN IV

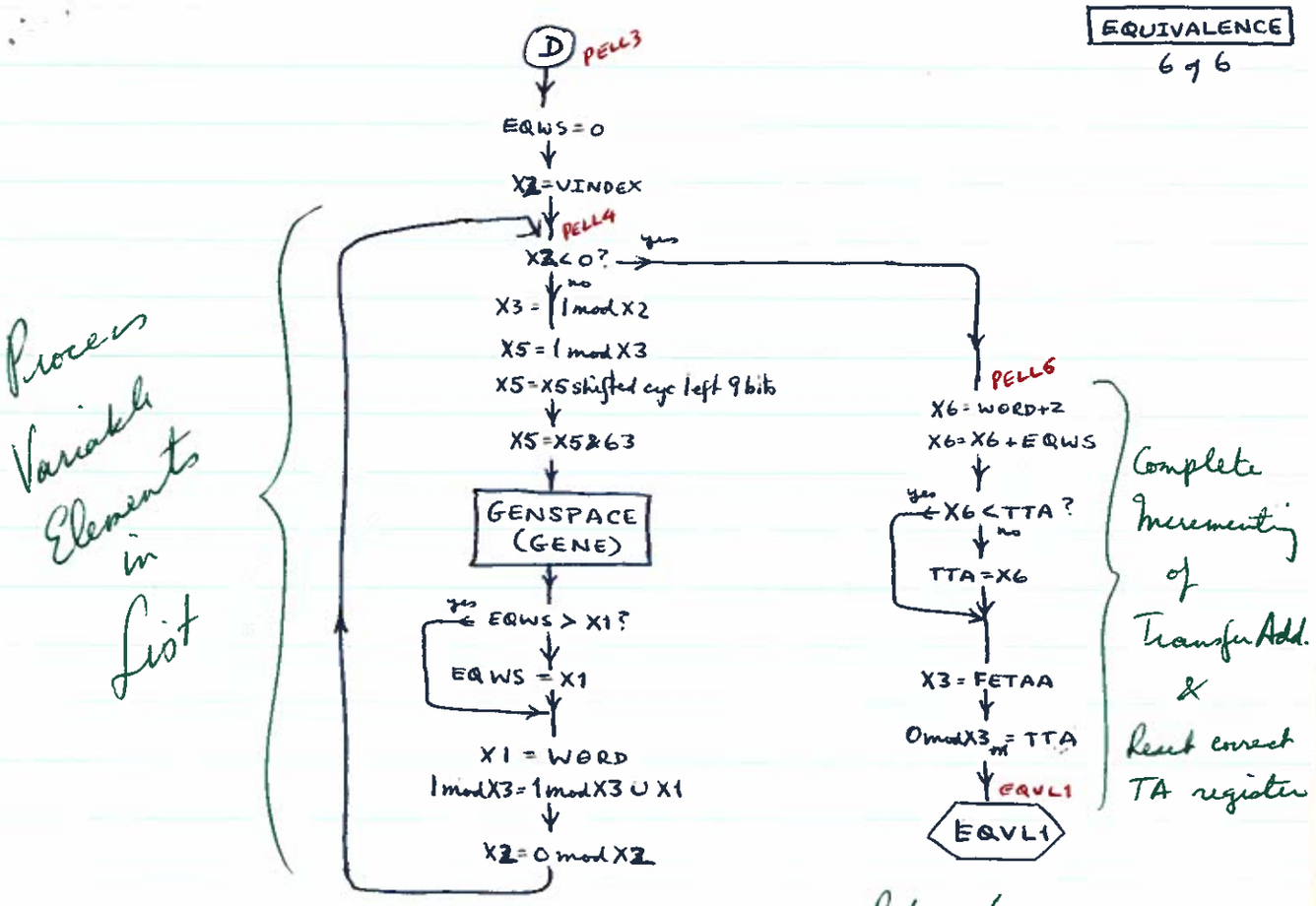
EQUIVALENCE

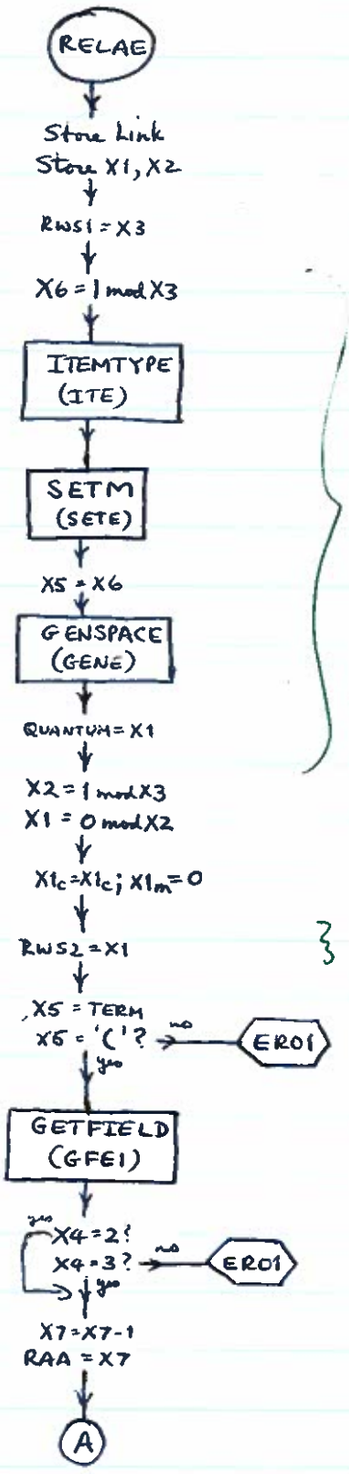
4 of 6



WORD	Pseudo Variable Data Word
WORD+1	-1.0 if direct 0 if indirect c/m if COMMON
WORD+2	Address relative to start of Area


C
PELE
REL=X8
X3=WORD+1
X3 = 1+X3m
X7 = SPRE?
X3 = 'VATA'
X7 = 'VREL'?
X3 = 'CNTA'
X7 = 'CREL'?
X3 = 'UVTA'
X7 = 'UVRE'?
X3 = 'P2TA'
X7 = 'P2RE'?
ER14
PELL1
FETAA=X3
X5 = 0 mod X3
TTA = X5m
FETAA = address of Correct TA register
TTA = contents of Correct TA register
X3 = AINDEX
X3 < 0?
AINDEX < 0 if no Arrays
X3 = WORD+1
X3 < 0?
ER10
Equivalencing Array & Direct Variable
X3 = 0?
CLIB (CLB5)
PELL2
X2 = AINDEX
X6, AINDEX = 0 mod X2
Look for Variable
D
X2 < 0?
Branch if no more arrays
PELL3
X6 = 2 mod X2
X6 = (WORD+2) - X6
X6 < 0?
ER11
ATA = address of (1,1,1)
ATA = X6
X3 = 3 mod X2
X6 = 1 mod X3
ITEMTYPE (ITE)
Error if already assigned in some Equivalence list
X1 = 2?
ER12
Complete the Header
MALI (MALE)
Complete Prog if COMMON or DATA
X1 = WORD+1
X7 = REL
X7 = 'P2EE'?
X1 = -1 . 0
2 mod X2 = X1
Process Array Element in list
Increment TTA if req'd.
X6 = ATA
X6 < TTA?
X6 - ATA
TTA = X6


D
PELL3
EQWS=0
X2=VINDEX
PELL4
X2<0?
yes
no
X3=1 mod X2
X5=1 mod X3
X5=X5 shifted cyc left 9 bits
X5=X5&63
GENSPACE (GENE)
yes
EQWS > X1?
EQWS=X1
X1=WORD
1 mod X3=1 mod X3 ∪ X1
X2=0 mod X2
PELL6
X6=WORD+2
X6=X6+EQWS
yes
X6<TTA?
no
TTA=X6
X3=FETAA
0 mod X3=TTA
EQVL1
EQVL1
Process Variable Elements in List
Complete Incrementing of TransferAdd. & Reset correct TA register
Return to process next Equivalence List.



For SpaceQuantum for an array element

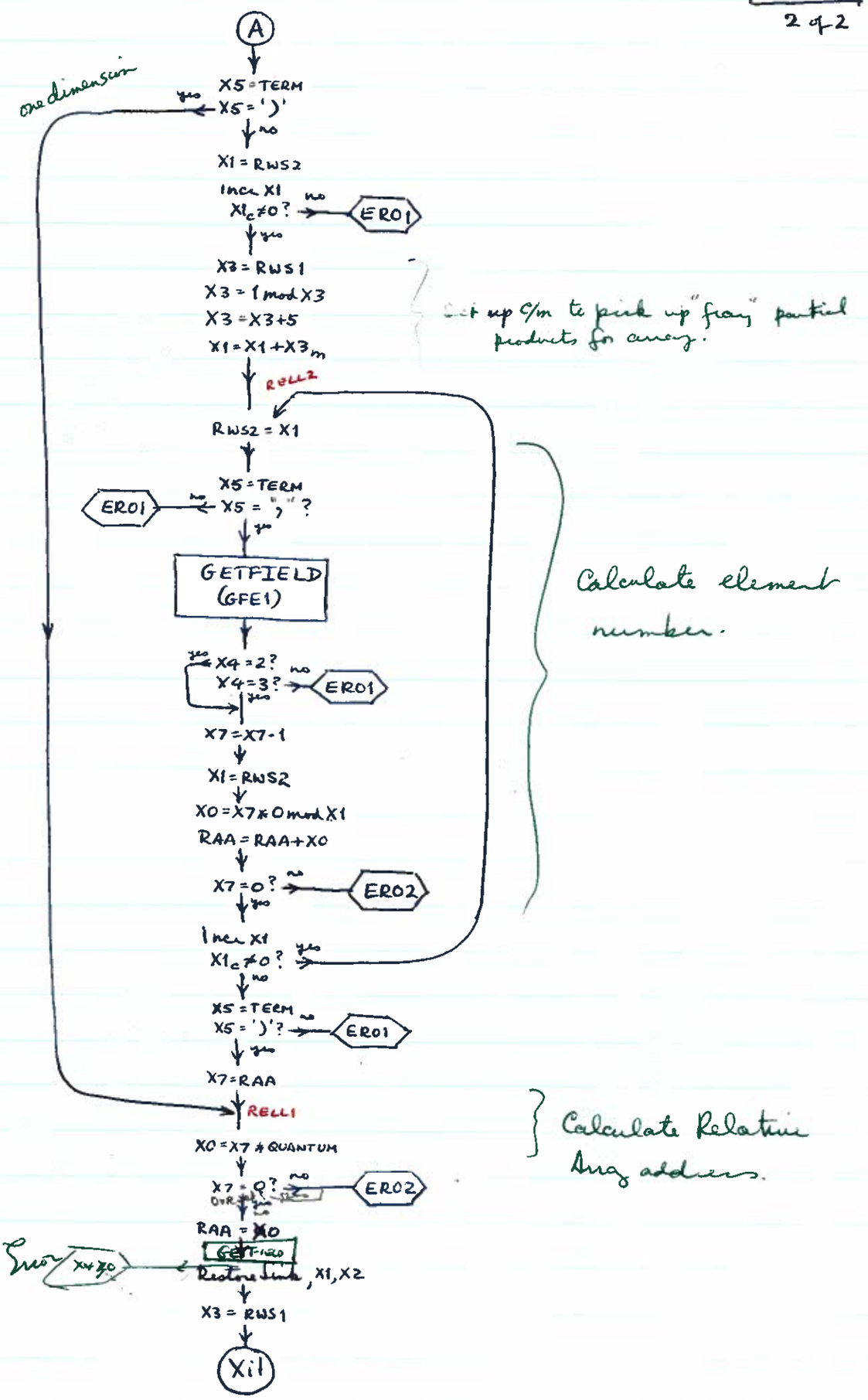
$X1_c$ = no of dimensions

Look at first dimension

FORTRAN IV

20/5/65

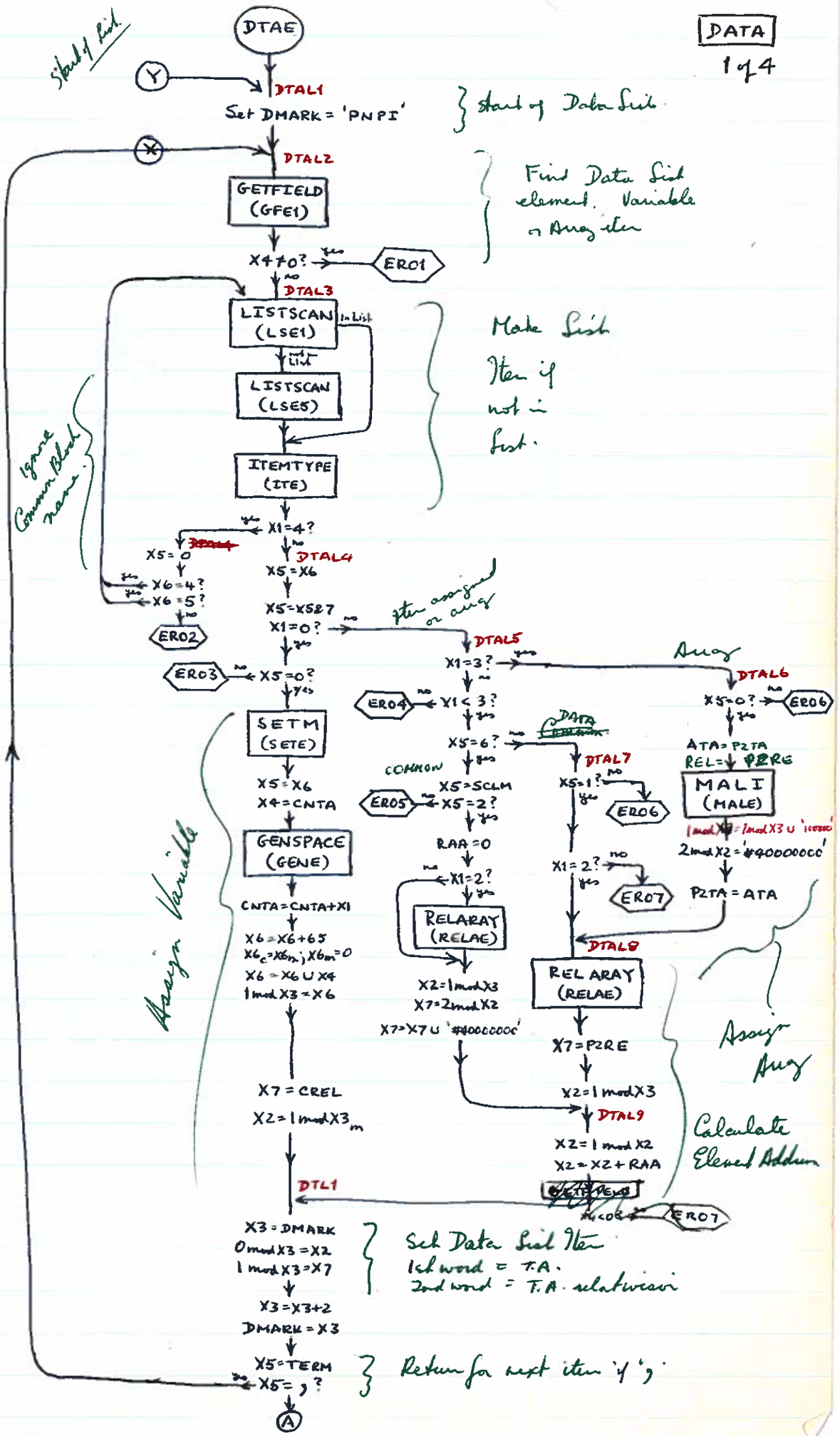
RELARAY
2 of 2

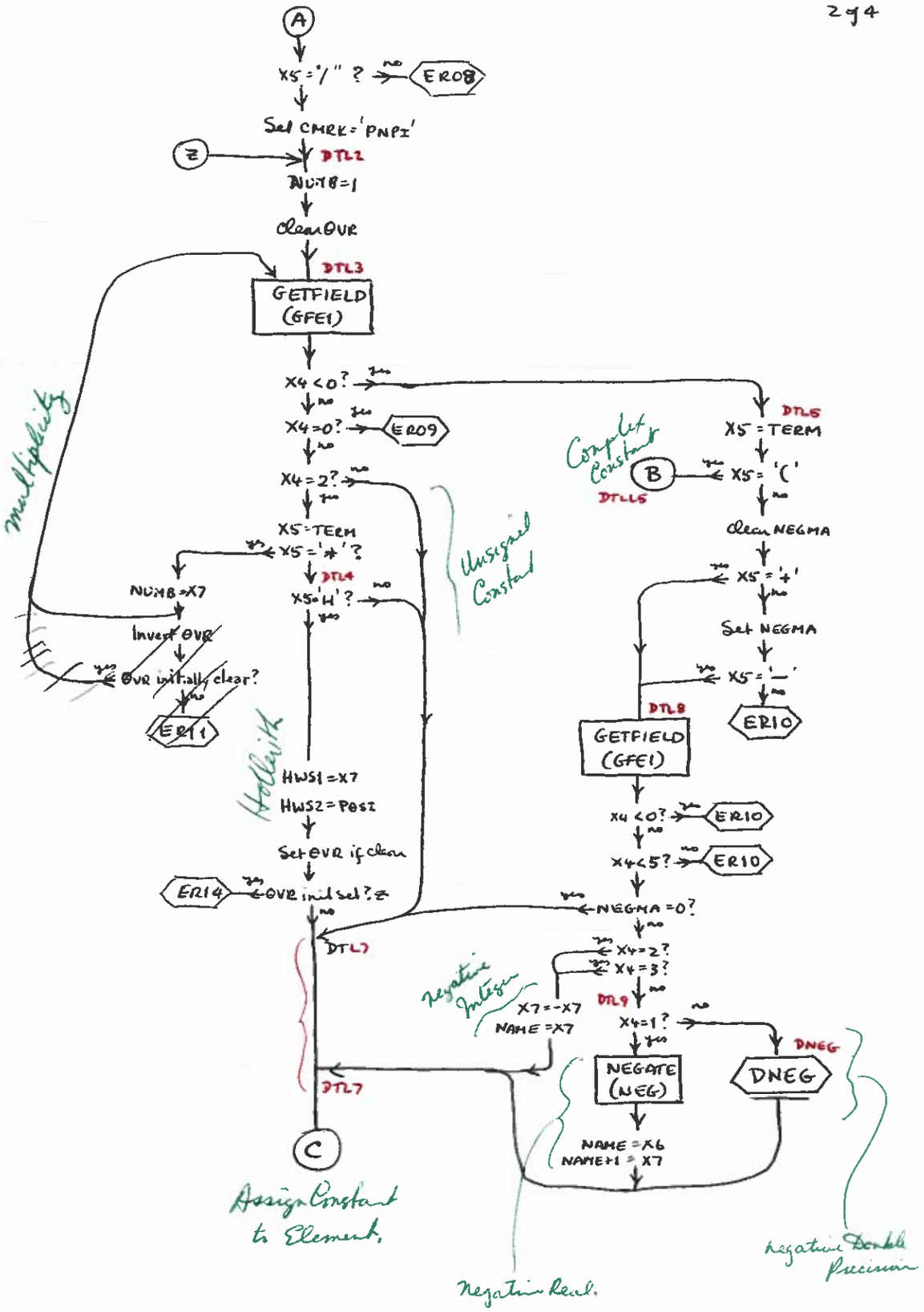


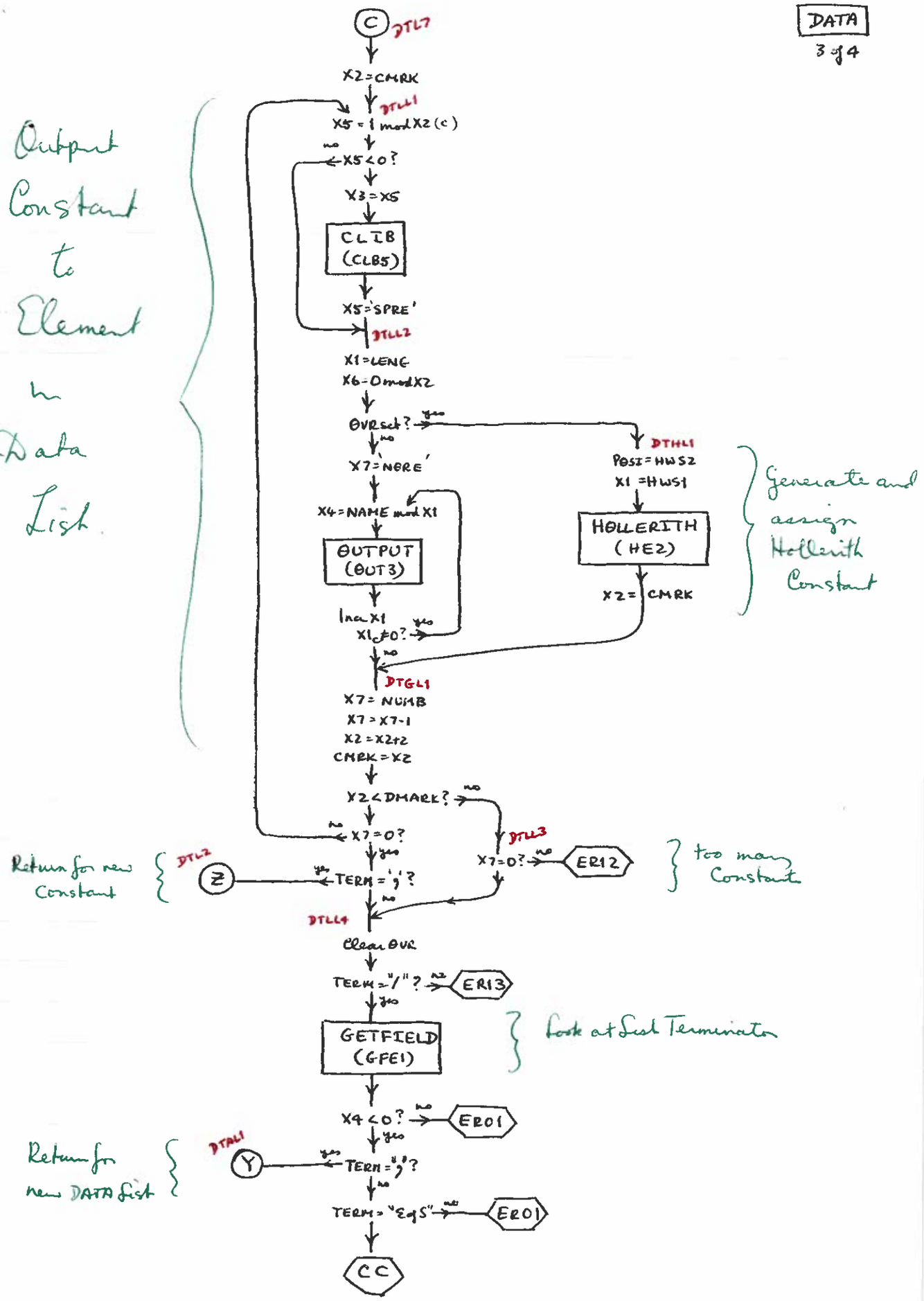
10/5/65

FG

DATA
194




A
X5 = '/' ?
no
ER08
Set CMRK = 'PNPI'
2
DTL2
NUMB = 1
Clear OVR
DTL3
GETFIELD (GFE1)
X4 < 0?
yes
X4 = 0?
yes
ER09
X4 = 2?
no
yes
X5 = TERM
X5 = '+' ?
yes
DTL4
NUMB = X7
Invert OVR
OVR initially clear?
yes
no
ER11
X5 = 'H' ?
no
yes
HWS1 = X7
HWS2 = POS1
Set OVR if clean
ER14
yes
OVR ind set?
no
DTL7
C
Assign Constant to Element.
Multiplicity
Unsigned Constant
Hollerith
DTL5
X5 = TERM
B
X5 = '('
Clear NEGMA
X5 = '+'
no
Set NEGMA
X5 = '-'
yes
ER10
DTL8
GETFIELD (GFE1)
X4 < 0?
yes
ER10
X4 < 5?
yes
ER10
NEGMA = 0?
yes
no
X4 = 2?
yes
X4 = 3?
yes
DTL9
X7 = -X7
NAME = X7
X4 = 1?
no
yes
NEGATE (NEG)
NAME = X6
NAME+1 = X7
DNEG
DNEG
DTL7
Complex Constant
Negative Integer
Negative Real.
Negative Double Precision



Output Constant to Element in Data List

Generate and assign Hollerith Constant

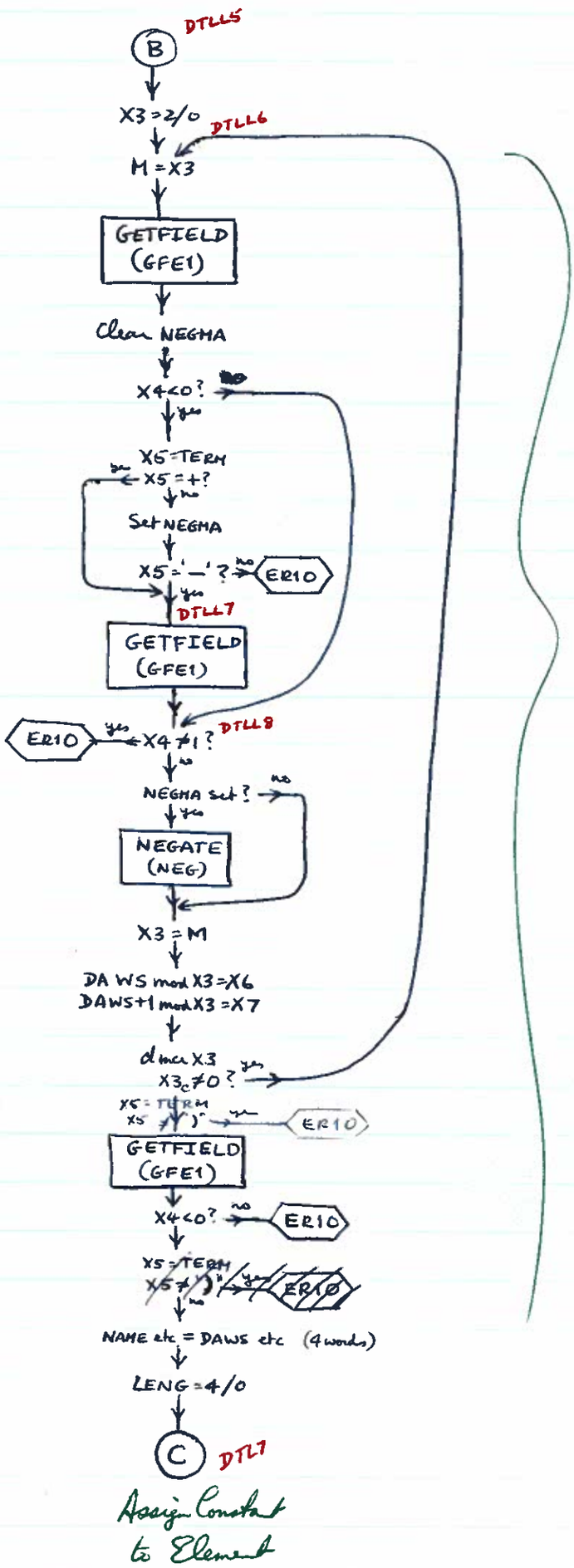
Return for new Constant

too many Constant

Look at List Terminator

Return for new DATA list

End of DATA Statement

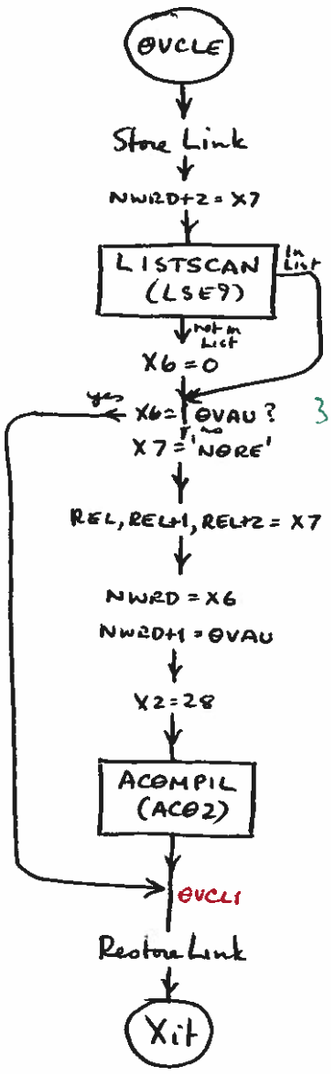


Form Complex Constant.

([±]a,[±]b)

Negate if required.

Assign Constant to Element



On entry. X7 = No. of words in regular call

X6 = 0 if called routine in 'Permanent' not an overlay call if in same area/unit.

Overlay Call

NWRD = A/U_B

NWRD+1 = A/U_A

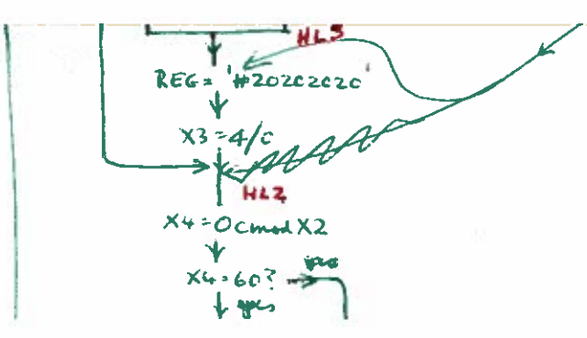
NWRD+2 = n

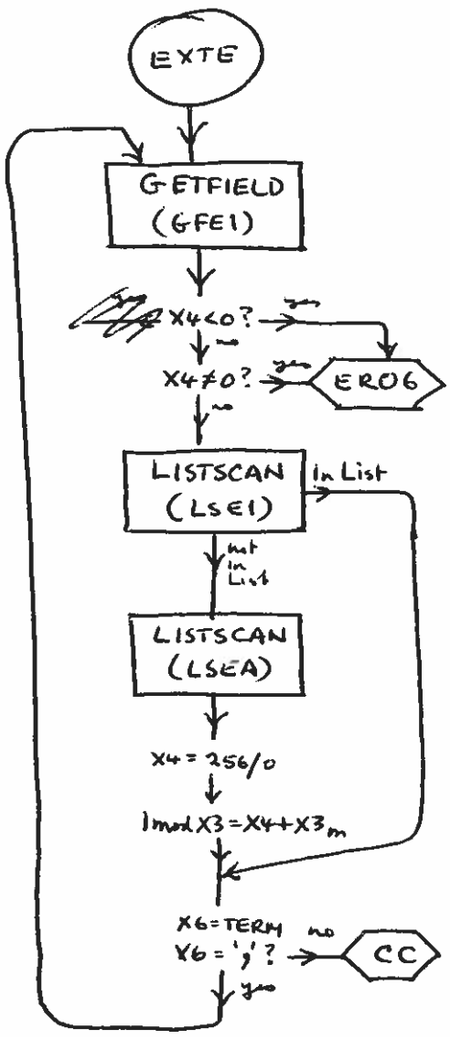
for call of form

CALL f (2,P0VL
A/U_B
A/U_A
n

regular call!

} Compile Overlay Calling Sequence





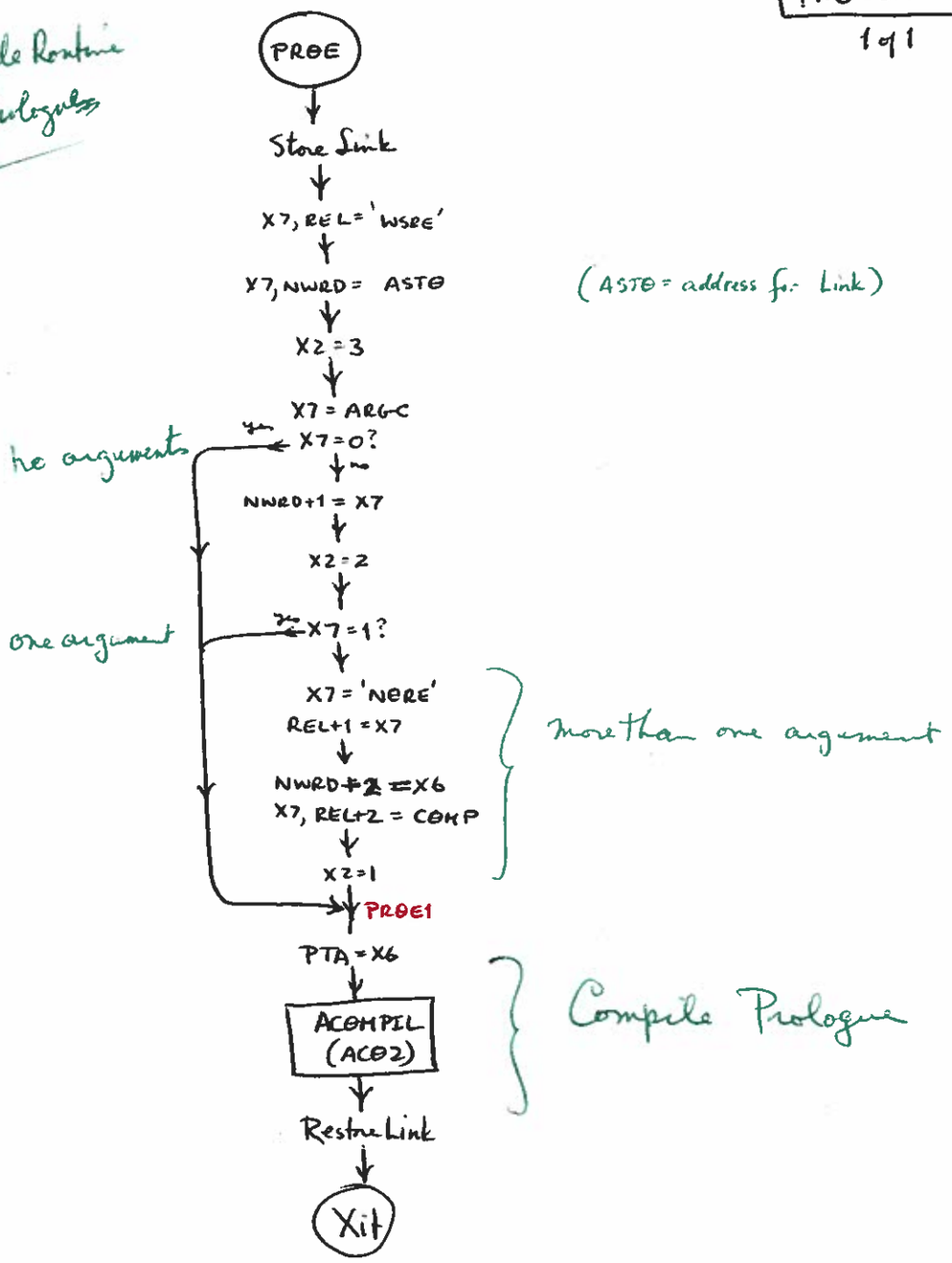
Get External Item

Error if not Alpha Field

Form Cue type field Item if not in List

Exit if field termination is not ';'

Compile Routine Prologue



(ASTO = address for Link)

ITEMTYPE

111