"EPILOG[400,400]"

A USERS' GUIDE TO THE DEC 10 PROLOG SYSTEM.

NOTE:THIS GUIDE IS UNDER DEVELOPMENT AND IS INCOMPLETE. COMMENTS ARE WELCOMED.

0. CONTENTS

1. HOW TO OBTAIN THE PROLOG SYSTEM

INITIATE A PROLOG SESSION USING THE STANDARD INITIAL STATE OF PROLOG WHICH IS HELD IN THE FILE SV1.50[400,400]. IE. PERFORM THE MONITOR COMMAND:-

```
.COPY OLD.PL=SV1.50[400,400]
```

WHEN INITIATING THE FIRST PROLOG SESSION (SEE BELOW).

2. HOW TO INITIATE A PROLOG SESSION

PERFORM THE FOLLOWING MONITOR COMMANDS:-

```
.COPY OLD.PL=#1                    ]:#1 IS THE NAME OF SOME SAVED STATE
                                    OF PROLOG.

.COPY TEXT.PL=#2                   ]:#2 IS THE NAME OF SOME FILE OF PROLOG
                                      SOURCE TEXT WHICH HAS ALREADY BEEN
                                      CREATED VIA EG. CEDRIC OR TECO.

.RUN PL50[400,400]

-TTY-LIREFICHIER-TTY.              ]:COMPILES THE SOURCE TEXT AND RETURNS
                                      CONTROL TO THE USER'S TERMINAL.

.                                  ]
.                                  ]:SEQUENCE OF PROLOG COMMANDS.
.                                  ]
```

```
-SAUVE.                          ]:SAVES THE NEW STATE OF PROLOG IN THE
                                   FILE NEW.PL OR
                                   --
-STOP.                           ]:MERELY HALTS THE PROLOG SESSION.
```

3. PROLOG SYNTAX

3.1 TO SYMBOL LEVEL

NOTE:THIS PART OF THE SYNTAX DEFINES A PROLOG PROGRAM AS A LIST OF SYMBOLS, EACH OF WHICH COMPRISES ONE OR MORE (ASCII) CHARACTERS.TWO SYMBOLS MAY BE SEPARATED BY ONE OR MORE SPACE CHARACTERS. CERTAIN SYMBOLS, WHEN THEY ARE CONSECUTIVE, MUST BE SEPARATED BY AT LEAST ONE SPACE CHARACTERTO AVOID AMBIGUITY (EG. TWO CONSECUTIVE IDENTIFIERS). UP TO 69 CHARACTERS MAY OCCUR ON A LINE. EACH LINE IS EXTENDED TO 69 CHARACTERS WITH THE APPROPRIATE NUMBER (ZERO OR MORE) OF SPACES. "NEW LINES" ARE OTHERWISE IGNORED.

```
'X' ←            ...THE SYMBOL COMPRISING THE CHARACTERS X.
[X] ←            <X!>
<X>-LIST ←       <X> [<X>-LIST]

COMMAND ←        CLAUSE
PROGRAM ←        CLAUSE-LIST '+' 'FIN' '.'
CLAUSE ←        <'*' COMMENT '.'
                 !OPERATOR-DECLARATION '.'
                 !LITERAL-LIST '.'>
OPERATOR-DECLARATION ←
                 '+' <T>-PRECEDENCE-<N>-OPERATOR
                    '(' DIRECTION_<T> ',' PRECEDENCE-<N> ')'
<T>-PRECEDENCE-<N>-OPERATOR ← IDENTIFIER
DIRECTION-LEFTRIGHT ← 'GD'
DIRECTION-RIGHTLEFT ← 'DG'
PRECEDENCE_<N> ← ...THE DIGIT N WHERE NOT N=Ø.
LITERAL ←        <'+'!'-'> ATOM
ATOM ←          <VARIABLE
                 !IDENTIFIER [ '(' TERMS ')' ]>
VARIABLE ←       '*' IDENTIFIER
TERMS ←          TERM [ ',' TERMS ]
TERM ←           TERM1
TERM-<N> ←       <[TERM-<N>] LEFTRIGHT-PRECEDENCE-<N>-OPERATOR TERM-<N+1>
                 !TERM-<N+1> RIGHTLEFT-PRECEDENCE-<N>-OPERATOR [TERM-<N>]
                 !TERM-<N+1>>
TERM-1Ø ←        <ATOM
                 !'(' TERM ')'
                 !STRING>
```

3.2 BELOW SYMBOL LEVEL

NOTE:THIS PART OF THE SYNTAX DEFINES EACH PROLOG SYMBOL IN TERMS OF STRINGS OF (ASCII) CHARACTERS.

```
COMMENT ←        COMMENT-CHARACTER-LIST
COMMENT-CHARACTER ← ...ANY CHARACTER EXCEPT '.'.
IDENTIFIER ← <ALPHANUMERIC-CHARACTER-LIST
```

```
              !UNRESERVED-CHARACTER>
ALPHANUMERIC-CHARACTER ← <LETTER!DIGIT>
STRING ←       '"' STRING-TOKEN-LIST '"'
STRING-TOKEN ←
              <'""'
              !NON-QUOTE-CHARACTER>
NON-QUOTE-CHARACTER ← ...ANY CHARACTER EXCEPT '"'.
CHARACTER ←   <LETTER
              !DIGIT
              !SPACE
              !SPECIAL-CHARACTER
              !UNRESERVED-CHARACTER>
LETTER ←      <'A'!'B'!...!'Z'>
DIGIT ←       <'Ø'!'1'!...!'9'>
SPACE ←        ' '
SPECIAL-CHARACTER ←
              <'('
              !')'
              !'*'
              !','
              !'"'>
UNRESERVED-CHARACTER ←
              <'['
              !'<'
              !'>'
              !'/'
              !'\'
              !'←'
              !']'
              !'+'
              !'-'
              !'.'
              !':'
              !';'
              !'?'
              !'!'
              !'$'
              !'%'
              !'='
              !'#'
              !'&'
              !'@'
              !APOSTROPHE>
APOSTROPHE ←   ...THE CHARACTER "'".

NOTE:THE SET OF UNRESERVED CHARACTERS IS IMPLEMENTATION-DEFINED.
```

4. A NOTE ON PROLOG SOURCE FILES AND THEIR AMENDMENT

CLAUSES FOR THE SAME PREDICATE (IE. CLAUSES WITH FIRST LITERAL HAVING THE SAME SIGN AND PREDICATE SYMBOL) SHOULD BE CONTIGUOUS IN THE PROGRAM. IF THEY ARE NOT CONTIGUOUS, THE LATER CLAUSES WILL "OVERWRITE" THE PRECEDING CLAUSES FOR THE SAME PREDICATE.

THE REASON FOR THIS IS TO ENABLE PREVIOUSLY COMPILED PROGRAMS TO BE AMENDED WITHOUT COMPLETE RE-COMPILATION. (COMPILATION IS RATHER SLOW SINCE THE "COMPILER" IS ITSELF WRITTEN IN PROLOG AND INTERPRETED IN THE STANDARD WAY.)

TO AMEND A PROGRAM:-

(1) EDIT THE ORIGINAL SOURCE FILE.

(2) COPY INTO FILE 'TEXT.PL' ONLY THE CLAUSES FOR PREDICATES FOR WHICH A CLAUSE HAS BEEN ADDED OR MODIFIED AND ALSO THE TERMINATING LINE '+FIN.'.

(3) COPY INTO FILE 'OLD.PL' THE PROLOG STATE FOR THE PREVIOUSLY COMPILED VERSION.

(4) EXECUTING THE PROLOG COMMAND '-TTY-LIREFICHIER-TTY.' WILL NOW PRODUCE A COMPILED VERSION OF THE AMENDED PROGRAM; HOWEVER, THE STORAGE OCCUPIED BY THE "OVERWRITTEN" CLAUSES WILL NOT BE RECLAIMED - TO AVOID THIS ONE SHOULD EVENTUALLY RE-COMPILE FROM SCRATCH.

5. THE PROLOG PROOF PROCEDURE

```
LET THE PROGRAM COMPRISE THE LIST OF CLAUSES C[1]C[2]...C[N];
LET BOOLEAN PROCEDURE REFUTE(W) BE
   (LET W COMPRISE THE LIST OF LITERALS P[1]P[2]..P[S];
    IF S=Ø THEN RETURN TRUE;
    FOR I TAKING SUCCESSIVE VALUES FROM 1 TO N DO
       (LET C[I] COMPRISE THE LIST OF LITERALS L[1]L[2]...L[R];
        IF THE SIGN OF L[1] IS OPPOSITE TO THE SIGN OF P[1] AND THE
        ATOMS OF L[1] AND P[1] ARE UNIFIABLE UNDER SUBSTITUTION $
        THEN
           (LET REFUTE((L[2]...L[R]P[2]...P[S])$) RETURN V;
            IF V=TRUE THEN RETURN TRUE;
            IF V=FALSE THEN CONTINUE
           )
       )
    RETURN FALSE
   );
LET THE COMMAND BE C[Ø];
LET REFUTE(C[Ø]) RETURN V;
IF V=TRUE THEN STOP;
IF V=FALSE THEN (OUTPUT '?';STOP).
```

NOTE: IN FACT THE IMPLEMENTATION OF PROLOG PARTITIONS THE PROGRAM INTO LISTS OF CLAUSES HAVING FIRST LITERAL WITH THE SAME PREDICATE AND SIGN AND MERELY ITERATES THROUGH THE APPROPRIATE LIST IN THE PARTITION INSTEAD OF THE ITERATION FROM 1 TO N ABOVE.

NOTE: AFTER EACH LITERAL IN A COMMAND THERE IS AN IMPLICIT '-/' TO PREVENT BACKTRACKING. [SEE SECTION 6]. FOR EXAMPLE THE COMMAND:-

```
-P-Q-R.
```

BEHAVES EXACTLY THE SAME AS:-

```
-P-/-Q-/-R-/.
```

6. EVALUABLE PREDICATES

A NUMBER OF EXTRA FACILITIES ARE PROVIDED BY "EVALUABLE PREDICATES". INSTEAD OF ATTEMPTING TO MATCH SUCH NEGATIVE LITERALS AGAINST POSITIVE

LITERALS IN THE PROGRAM, THE PROLOG INTERPRETER EXECUTES A CALL TO AN APPROPRIATE SUBROUTINE. TYPICALLY,EVALUABLE PREDICATES CAUSE SIDE EFFECTS.

A "CALL" TO AN EVALUABLE PREDICATE
EITHER (1) SUCCEDS - ANALAGOUS TO A SUCCESSFUL MATCH TO A UNIT CLAUSE,
OR (2) FAILS - ANALAGOUS TO A FAILURE TO MATCH, INDUCING BACK-TRACKING,
OR (3) IS AN ERROR (IF THE ARGUMENTS ARE INAPPROPRIATE) - LEADING TO AN ERROR MESSAGE OR UNDEFINED RESULTS.

6.0 SUMMARY OF THE EVALUABLE PREDICATES AVAILABLE

INPUT OUTPUT

-LU	-TTY	-SORCHA	-SAUVE
-LUB	-LIREFICHIER	-SORTER	-STOP
-ECRIT	-BOOLISTE		
-LIGNE	-IMPRIME		

ARITHMETIC AND CHARACTERS

-PLUS	-LETTRE
-MOINS	-CHIFFRE
-MULT	-BLANC
-DIV	-ETOILE
-RESTE	-VIRG
	-PARG
-INF	-PARD
	-GUILLEMET

EXTRA CONTROL

- -/
- -/(TERM)
- -ANCETRE
- -ETAT

DATABASE MANIPULATION AND META-PREDICATES

-AJOUT	-*X
-AJOUTB	-UNIV
-AJOUTC	-ATOME
-SUPP	-EGALF

6.1 INPUT OUTPUT

-LU(*C)

A SINGLE CHARACTER IS READ FROM THE CURRENT INPUT STREAM AND UNIFIED WITH THE TERM *C.

-LUB(*C)

THE FIRST NON-BLANK CHARACTER READ FROM THE CURRENT INPUT STREAM IS UNIFIED WITH THE TERM *C.

-ECRIT(*C)

THE TERM *C MUST BE A SINGLE CARACTER, WHICH IS WRITTEN TO THE OUTPUT STREAM.

-LIGNE

THE CHARACTERS IN THE OUTPUT BUFFER ARE OUTPUT TO THE TELETYPE ON A NEW LINE.

-TTY

THE CURRENT INPUT STREAM IS SWITCHED FROM THE TELETYPE TO THE FILE 'TEXT.PL' OR VICE VERSA.

-LIREFICHIER

CLAUSES ARE READ FROM THE CURRENT INPUT STREAM AND ADDED TO THE DATABASE, TERMINATING WHEN '+FIN.' IS FOUND.

-BOOLISTE

CAUSES A BOOLEAN SWITCH WHICH CONTROLS LISTING TO BE SWITHED FROM OFF TO ON OR FROM ON TO OFF. WHEN THE SWITCH IS ON, EACH CLAUSE READ IS LISTED ON THE TERMINAL.

-IMPRIME

CAUSES THE CURRENT LINE IN THE INPUT STREAM TO BE LISTED ON THE TERMINAL, PROVIDED IT HASN'T ALREADY BEEN LISTED BY 'BOOLISTE'. (?)

-SORCHA(*S)

*S MUST BE A CHARACTER STRING (ENCLOSED IN DOUBLE QUOTES ("")), THE CHARACTERS ARE WRITTEN TO THE OUTPUT STREAM.

-SORTER(*X)

THE TERM *X IS WRITTEN TO THE OUTPUT STREAM, WITH THE APPROPRIATE FORMAT FOR EXPRESSIONS INVOLVING OPERATORS.

-SAUVE

THE CURRENT STATE OF PROLOG IS WRITTEN TO THE FILE 'NEW.PL' AND THE SESSION IS TERMINATED.

-STOP

THE CURRENT SESSION IS TERMINATED. (QUICKER TO CONTROL C).

6.2 ARITHMETIC AND CHARACTERS

-PLUS(*N1,*N2,*N3)

*N1 AND *N2 MUST BE INTEGERS AND *N3 EITHER AN INTEGER OR A VARIABLE. *N3 IS UNIFIED WITH THE SUM OF *N1 AND *N2.

-MOINS(*N1,*N2,*N3)

SAME CONDITIONS ON *N1,*N2,*N3. *N3 IS UNIFIED WITH THE DIFFERENCE OF *N1 AND *N2.

-MULT(*N1,*N2,*N2)

SAME CONDITIONS ON *N1,*N2,*N3. *N3 IS UNIFIED WITH THE PRODUCT OF *N1 AND *N2.

-DIV(*N1,*N2,*N3)

SAME CONDITIONS ON *N1,*N2,*N3. *N3 IS UNIFIED WITH THE INTEGER PART OF *N1 / *N2.

-RESTE(*N1,*N2,*N3)

SAME CONDITIONS ON *N1,*N2,*N3. *N3 IS UNIFIED WITH THE INTEGER REMAINDER OF *N1 / *N2.

-INF(*X,*Y)

IT IS AN ERROR IF EITHER ARGUMENT IS NEITHER AN INTEGER NOR A SINGLE CHARACTER. THE CALL SUCCEEDS IF *X IS STRICTLY LESS THAN *Y ACCORDING TO THE NATURAL ORDER FOR INTEGERS OR THE E.B.C.D.I.C. CODE FOR CHARACTERS. ANY NON-DIGIT CHARACTER IS LESS THAN ANY INTEGER.

-LETTRE(*C)

*C MUST BE A SINGLE CHARACTER. THE CALL SUCCEEDS IF *C IS A LETTER.

-CHIFFRE(*C)

*C MUST BE A SINGLE CHARACTER. THE CALL SUCCEEDS IF *C IS A DIGIT.

THE FOLLOWING PREDICATES UNIFY VARIOUS SPECIAL CHARACTERS AGAINST THE ARGUMENT *C :-

```
-BLANC(*C)      :' '
-ETOILE(*C)     :'*'
-VIRG(*C)       :','
-PARG(*C)       :'('
-PARD(*C)       :')'
-GUILLEMET(*C)  :'"'
```

6.3 EXTRA CONTROL

-/

SUPPRESSES ANY FURTHER CHOICES FOR ANY CHOICE POINTS SINCE AND INCLUDING THE POINT AT WHICH THE PARENT LITERAL WAS MATCHED AGAINST THE DATABASE. I.E IF BACKTRACKING RETURNS TO THIS POINT THE RESULT IS TO FAIL THE PARENT LITERAL. FOR EXAMPLE, IF AN INSTANCE OF CLAUSE:-

```
+P-A-B-C.
```

CAUSES A CALL TO CLAUSE:-

```
+B-D-E-/-F-G.
```

AND IF CONTROL EVENTUALLY REACHES -F AND FAILS TO GET A MATCH FOR -F THEN THE NEXT ACTION WILL BE TO TRY TO FIND A DIFFERENT MATCH FOR THE LAST CHOICE MADE IN THE SUB-PROOF FOR -A.

-/(*L)

IT IS AN ERROR IF *L ISN'T A TERM DENOTING A LITERAL, IE. A TERM OF THE FORM +(*X) OR -(*X) WHERE *X MAY BE ANY TERM. THE CALL FAILS IF *L CAN'T BE UNIFIED WITH ANY OF THE ANCESTORS OF THE -/(*L) LITERAL. OTHERWISE *L IS UNIFIED AGAINST THE MOST RECENT POSSIBLE ANCESTOR AND ANY CHOICES MADE SINCE AND INCLUDING THE TIME THE ANCESTOR WAS MATCHED ARE "FROZEN" AS ABOVE. [THE ANCESTORS OF A LITERAL ARE ITS PARENT AND ITS PARENT'S ANCESTORS.] FOR EXAMPLE THE SEQUENCE:-

```
-/(-(SUBGOAL(*X))-FAIL
```

WILL CAUSE THE SEARCH FOR A PROOF OF THE MOST RECENT SUBGOAL OF THE FORM '-SUBGOAL(*X)' TO FAIL PROVIDED '-FAIL' FAILS.

-ANCETRE(*L)

*L MUST BE A TERM DENOTING A LITERAL. *L IS UNIFIED WITH THE MOST RECENT ANCESTOR FOR WHICH THIS IS POSSIBLE, IF ANY.

-ETAT(*S)

*S IS UNIFIED WITH A TERM REPRESENTING THE CURRENT STATE OF THE PROOF. THIS TERM IS A LIST OF LISTS OF LITERALS. THE FIRST MEMBERS OF EACH SUBLIST ARE THE ANCESTORS OF THE 'ETAT' LITERAL IN ORDER, WITH THE MOST RECENT FIRST. THE REMAINING LITERALS IN EACH SUBLIST ARE THE REMAINING LITERALS IN THE CLAUSE WHICH WAS MATCHED AGAINST THE CORRESPONDING ANCESTOR. FOR EXAMPLE GIVEN THE OPERATOR DECLARATIONS:-

```
+.(DG,5).
++(GD,6).
+-(GD,6).
```

AND THE PROGRAM:-

```
+FACT(Ø,1)-ETAT(*S)-SORTER(*S)-LIGNE.
+FACT(*X,*Y)-MINUS(*X,1,*X1)-FACT(*X1,*Y1)-PROD(*X,*Y1,*Y).
```

THE STATE OUTPUT WILL BE:-

```
(-FACT(Ø,1).-SORTER(*S).-LIGNE.NIL).
(-FACT(1,*Y1).-PROD(1,1,*Y1).NIL).
(-FACT(2,*Y2).-PROD(2,*Y1,*Y2).NIL).
.
.
.
NIL
```

6.4 DATABASE MANIPULATION AND META-PREDICATES

-AJOUT(*C)

*C MUST BE A LIST OF TERMS DENOTING LITERALS (CF. ABOVE). *C IS INTERPRETED AS A CLAUSE WHICH IS ADDED TO THE DATABASE AT THE BEGINNING OF THE LIST OF CLAUSES WHOSE FIRST LITERAL HAS THE SAME SIGN AND PREDICATE.

-AJOUTB(*C)

AS FOR AJOUT, BUT THE NEW CLAUSE IS ADDED AT THE END OF THE LIST. IF THE LAST CLAUSE ADDED WAS NOT OF THE SAME PREDICATE AND SIGN, THE NEW CLAUSE REPLACES ANY OTHER CLAUSES IN ITS CORRESPONDING LIST.

-AJOUTC(*C)

AS FOR AJOUTB, BUT WITHOUT THE PROPERTY OF OVERWRITING.

-SUPP(*C)

*C MUST BE A TERM DENOTING A CLAUSE. IT IS UNIFIED WITH THE FIRST MEMBER OF THE LIST OF CLAUSES FOR THAT SIGN AND PREDICATE. IF THE UNIFICATION SUCCEEDS, THE CLAUSE IS DELETED.

-*X

NOTE THAT THE SYNTAX ALLOWS A VARIABLE TO APPEAR AS THE ATOM OF A LITERAL. THE TERM BOUND TO THAT VARIABLE IS INTERPRETED AS A ATOM.

-UNIV(*X,*Y)

THE "DECOMPOSITION" OF THE TERM *X IS UNIFIED WITH THE TERM *Y. IF THE FIRST TERM IS A VARIABLE IT IS UNIFIED WITH THE TERM WHOSE DECOMPOSITION IS *Y. FOR EXAMPLE, THE DECOMPOSITION OF:-

```
FOO(A,*X,FIE(*Y))
```

IS:-

```
(F.O.Ø.NIL).A.*X.FIE(*Y).NIL
```

WHERE '.' HAS BEEN DECLARED AS AN OPERATOR.

-ATOME(*X,*Y)

EXACTLY THE SAME AS UNIV EXCEPT WHERE *X IS A VARIABLE AND AND THE TERM CORRESPONDING TO *Y DOESN'T ALREADY EXIST IN THE DATABASE, IN WHICH CASE THE CALL FAILS.

-EGALF(*X,*Y).

THE CALL SUCCEEDS IF *X AND *Y ARE SYNTACTICALLY IDENTICAL TERMS.

A1. EXAMPLE OF A PROLOG SESSION

```
.COPY OLD.PL=SY1.5Ø[4ØØ,4ØØ]
.COPY TEXT.PL=PSORT

.RUN PL5Ø[4ØØ,4ØØ]
```

```
-BOOLISTE-TTY-LIREFICHIER-TTY-BOOLISTE.

 ***PSORT:A SORT PROGRAM IN PROLOG.

+.(DG,5).

+SORTED(*X.*LØ,*L)-SORTED(*LØ,*L1)-INSERT(*X,*L1,*L).
+SORTED(NIL,NIL).

+INSERT(*X,*Y.*LØ,*Y.*L)-INF(*Y,*X)-/-INSERT(*X,*LØ,*L).
+INSERT(*X,*LØ,*X.*LØ).

+SØRT(*L)-SORTED(*L,*L1)-SØRTER(*L1)-LIGNE.

+FIN.

↑C

.TIME
11.4Ø
11.4Ø
KILO-CØRE-SEC=475

.CON

-SORT(I.T.A.L.L.S.T.A.R.T.E.D.W.I.T.H.A.R.I.S.T.O.T.L.E.NIL).

A . A . A . D . E . E . H . I . I . I . L . L . L . O . R . R . S . S
 . T . T . T . T . T . T . W . NIL
↑C

.TIME
6.32
17.72
KILO-CORE-SEC=785

.CON

-STOP.


END OF EXECUTION
CPU TIME: 15.62 ELAPSED TIME: 5:17.7Ø
EXIT
```

A2. DIFFERENT VERSIONS OF THE SYSTEM

THE STANDARD PROLOG INTERPRETER PL5Ø RUNS IN 5ØK OF CORE. IF MORE CORE IS AVAILABLE (EG. IN THE EVENING), THERE IS ALSO A 75K VERSION PL75. UNFORTUNATELY, IT IS NOT POSSIBLE TO COMMUNICATE A 5ØK "STATE OF PROLOG" TO THE 75K INTERPRETER OR VICE VERSA. CONSEQUENTLY THERE HAS TO BE A 75K VERSION OF THE STANDARD INITIAL STATE CALLED SV1.75[4ØØ,4ØØ].

IN ADDITION TO THE ORIGINAL PROLOG SUPERVISER SV1 CONTAINED IN THE STANDARD INITIAL STATE, THERE IS ALSO A REVISED AND EXTENDED VERSION

SV2. AMONG OTHER THINGS, SV2 CORRECTS THE "BUG" DESCRIBED IN APPENDIX 3. UNFORTUNATELY SV2 IS CONSIDERABLY LARGER THAN SV1, LEAVING MUCH LESS ROOM FOR THE USER'S OWN PROGRAM. FOR FURTHER INFORMATION ON SV2, CONTACT DAVE WARREN.

TO SUMMARISE, THE DIFFERENT VERSIONS OF THE PROLOG SYSTEM COMPRISE THE FOLLOWING FILES, ALL ON [400,400]:-

```
PL50.SAV    :50K INTERPRETER,
PL75.SAV    :75K INTERPRETER,
SV1.50      :50K INITIAL STATE WITH ORIGINAL SUPERVISER,
SV1.75      :75K INITIAL STATE WITH ORIGINAL SUPERVISER,
SV2.50      :50K INITIAL STATE WITH NEW SUPERVISER,
SV2.75      :75K INITIAL STATE WITH NEW SUPERVISER.
```

A3. A "BUG" IN THE SYSTEM

THIS IS NOT SO MUCH A BUG AS A BUILT-IN DEFFECT ARISING FROM THE SYNTAX OF AN OPERATOR DECLARATION. IT MEANS THAT ONE MUST AVOID INPUTTING OR OUTPUTTING TERMS CONTAINING A FUNCTION F (OF ZERO OR MORE ARGUMENTS) WHICH IS SYNONYMOUS WITH AN ALREADY DEFINED PREDICATE OF TWO ARGUMENTS. THE CLAUSE(S) DEFINING SUCH A PREDICATE EG:-

```
+F(*X,*Y)-.....
```

WILL BE TREATED AS OPERATOR DECLARATIONS (!) WITH BIZARRE RESULTS.

A4. ANOTHER BUG

THERE SEEMS TO BE AN OBSCURE BUG WHEREBY CLAUSES FOR A TWO-ARGUMENT PREDICATE WHOSE NAME IS A SINGLE LETTER CORRUPT PRECEDING OPERATOR DECLARATIONS. EG.:-

```
+.(DG,1).
.
.
.
+W(*X,NIL)-.....
```

IT'S PROBABLY ADVISABLE TO AVOID SINGLE LETTER PREDICATE NAMES.