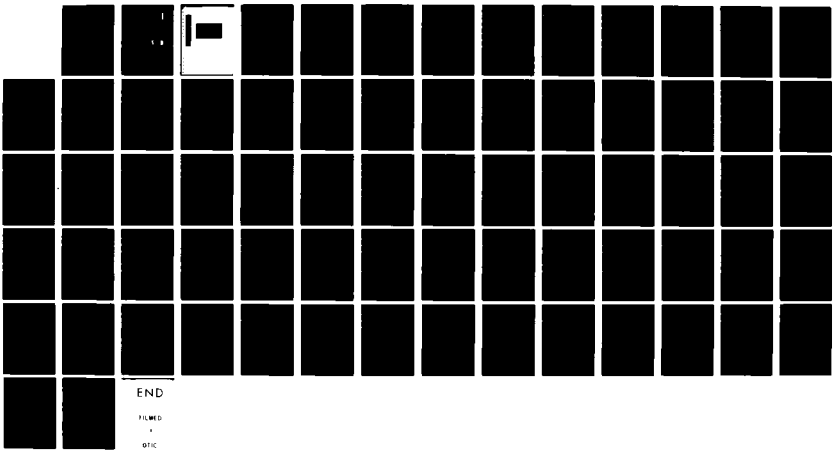
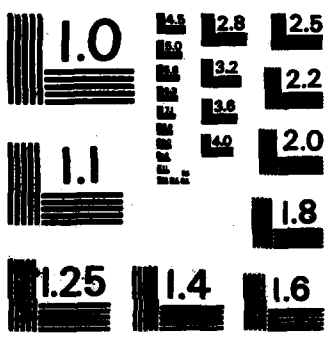
AD-A121 985 RIME: THE RECOVERABLE ITEM MANAGEMENT EVALUATOR VOLUME II SECTION I EVENTS GENERATOR PROGRAMS(U) DECISION SYSTEMS DAYTON OH W S DEMMY MAY 80 TR-80-02-A 1/1

UNCLASSIFIED F33600-78-C-0524 F/G 9/2 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

PHOTOGRAPH THIS SHEET

AD A121985

DTIC ACCESSION NUMBER

III

LEVEL

I

INVENTORY

RIME: Vol. II, Section I

DOCUMENT IDENTIFICATION

Rept. No. TR-80-02-A

Contract F33600-78-C-0524 May 80

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

DISTRIBUTION STATEMENT

ACCESSION FOR		
NTIS	GRA&I	☒
DTIC	TAB	☐
UNANNOUNCED		☐
JUSTIFICATION		
BY		
DISTRIBUTION /		
AVAILABILITY CODES		
DIST	AVAIL AND/OR SPECIAL	
A		

DISTRIBUTION STAMP

DTIC
ELECTE
DEC 1 1982
S D
D

DATE ACCESSIONED

DATE RECEIVED IN DTIC

PHOTOGRAPH THIS SHEET AND RETURN TO DTIC-DDA-2

DTIC FORM OCT 79 70A

DOCUMENT PROCESSING SHEET

RIME:

The Recoverable Item Management Evaluator:

Volume II, Section I

Events Generator Programs

RIME:

The Recoverable Item Management Evaluator:

Volume II, Section I

Events Generator Programs

by

W. Steven Demmy

May 1980

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

TR-80-02-A

DECISION SYSTEMS
2125 Crystal Marie Drive
Dayton, Ohio 45431

(513) 426-8515

82 11 30 025

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
RIME: The Recoverable Item Management Evaluator: Volume II: Program Listings and Narratives

5. TYPE OF REPORT & PERIOD COVERED
INTERIM
August 79-Jun 80

6. PERFORMING ORG. REPORT NUMBER
TR-80-02-A

7. AUTHOR(s)
W. Steven Demmy

8. CONTRACT OR GRANT NUMBER(s)
C-0524
F33600-78-~~R-0214~~

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Decision Systems
3575 Charlene Drive
Dayton, Ohio 45432

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
23041A5

11. CONTROLLING OFFICE NAME AND ADDRESS
2750th ABW/PMA
BLDG 1, Area C
Wright-Patterson AFB, Ohio

12. REPORT DATE
May 1980

13. NUMBER OF PAGES
298

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

A

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
Recoverable item, multi-echelon, inventory/repair, simulation, METRIC, MOD-METRIC, AFLC

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes the Recoverable Item Management Evaluator (RIME), a FORTRAN simulation model for evaluating the relative cost-effectiveness of analytic optimization procedures proposed for use in Air Force Logistics Command recoverable item management systems. Major features of the model include (a) the use of actual Air Force demand histories to drive the model demand processes, (b) modeling of current Air Force statistical estimation procedures, and (c) modeling of the dynamic interations among initial provisioning, replenishment and distribution policies. Volume II documents the programs for RIME.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

A

TABLE OF CONTENTS

SECTION	
I.	Introduction
II.	Events Generator Programs
III.	Levels Computation System Programs
IV.	RIME Simulation Model Programs
V.	Job Control Language Files

Section I.

Introduction

This is the second of two volumes describing the features of the Recoverable Item Management Evaluator (RIME). RIME is a FORTRAN-based model of a multi-echelon inventory system consisting of a depot level repair facility, possibly several operating bases and an aircraft overhaul facility. The model was specifically designed to evaluate proposed inventory management policies for the control of Air Force Logistics Command recoverable item spares. Volume I of this report describes the philosophy, organization, and input/output features of RIME. This volume documents the statement listings of each RIME program, and provides narratives describing the function of each of these routines.

This Volume contains four major sections. Section I documents programs used in the RIME Exogenous Events Generator, while Section II documents programs used in the Levels Computation System. Section III documents programs in the RIME Simulation Model. Finally, Section IV documents Job Control Language files required to execute the RIME System.

Within each section, routines are presented in alphabetical order. Documentation of each routine consists of a program narrative and a listing of the FORTRAN code for that routine. Additional comments are included as a part of the FORTRAN code for the more complex routines. Definitions of major RIME variables are documented in Volume I, and familiarity with the contents of Volume I is assumed.

Section I

Events Generator Programs

List of Programs

Program
ASSIGN
BASEDA
EVTGNB
EVTGN2
LRUEV2
NOSET1
RANBAS
SCHIND
SRUEVT
SRUIND
READF2
SRUPRT

Subroutine: ASSIGN

Function:

This routine determines the number of SRU failures to relate to the current LRU reparable generation event.

Calling Parameters:

LBRGN	=	Total number of LRU reparable generations remaining to be considered.
NOSRUF(K)	=	The total number of SRU K reparable generations remaining to be assigned to LRU reparable generations.
NOINL(K)	=	The number of SRU K reparable generations to be assigned to the LRU reparable generation currently be considered.

Description:

Routine ASSIGN first computes the total number of SRU failure exposures SRUEX. This equals the total number of LRU reparable generations remaining (LBRGN) times the Quantity Per Application (IQPA(K)) for SRU K. The probability a specific unit of SRU K of the current LRU has failed is then P=NOSRUF(K)/SRUEX. Subroutine ASSIGN obtains a uniform random number R by calling subroutine RANDU, and compares this number to the probability P. If R=P, we

assume that the current SRU is related to the LRU failure and NOINL(K) is incremented by one. This Monte Carlo process is then repeated for each SRU component contained in the LRU. At the conclusion of the routine, NOINL(K) contains the number of reparable generations of SRU K to be associated with the current LRU failure.

```
                SUBROUTINE ASSIGN(LBRGN,NOSRUF,NOINL)                    5
      COMMON/IWT/IWT(20)
      COMMON/NSRU/NSRU
      COMMON/IOPA/IOPA(1)
      DIMENSION NOINL(1),NOSRUF(1)
        IF(IWT(5).EQ.1) WRITE(6,17)
   17 FORMAT(' ********  ENTER ASSIGN  *********')
C
C               LOOP THROUGH EACH SRU(K) IN THE LRU, WHERE K=1,2,....,NSRU
C
      DO 800 K=1,NSRU
C
C               COMPUTE TOTAL SRU FAILURE EXPOSURES
C
      SRUEX=FLOAT(LBRGN*IOPA(K))
C
C               INITIALIZE TO ZERO THE COUNTER OF THE NUMBER OF
C             SRU(K) UNITS THAT HAVE FAILED IN THIS LRU.
C
      NOINL(K)=0
C
C           LOOK THROUGH EACH UNIT KK OF SRU(K) INSTALLED
C             IN THE LRU, WHERE KK=1,2,....,IORT(K)
C
       KKK=IOPA(K)
       DO 300 KK=1,KKK
C
C               COMPUTE THE FAILURE PROBABILITY P FOR THIS UNIT
C
      P=FLOAT(NOSRUF(K))/SRUEX
C
C               MONTE CARLO TO DETERMINE IF UNIT KK IS TO BE
C               RELATED TO THE LRU FAILURE.
C
      R=RANDU(0,2)
C
C               IF R LESS THAN OR EQUAL TO P, SET IFLAG=1 TO
C               INDICATE THAT THIS SRU IS RELATED TO THE
C               LRU FAILURE; OTHERWISE, SET IFLAG=0.
C
      IFLAG=0
      IF(R.LE.P) IFLAG=1
C
       IF(IWT(5).EQ.1)WRITE(6,19)"ASSIGN, SRUEX,  KK,    P,    R, IFLAG"
   19 FORMAT(V)
      IF(IWT(5).EQ.1)WRITE(6,397)SRUEX,KK,P,R,IFLAG
  397 FORMAT(T10,F5.0,I5,2F8.3,I5)
C
C               IF IFLAG EQUALS 1, DECREMENT THE SRU'S FAILURES
C               COUNTER BY 1 AND INCREASE COUNTER FOR THE
C               TOTAL SRU(K)UNITS THAT HAVE FAILED IN THIS
C               LRU, OTHERWISE GO TO NEXT STEP
```

ASSIGN

```
C
      IF(IFLAG.EQ.1) NOSRUF(K)=NOSRUF(K)-1
      IF(IFLAG.EQ.1) NOXML(K)=NOXML(K)+1
C
C                 DECREMENT THE EXPOSURES COUNTER.
C
      SRUEX=SRUEX-1
C
C               END OF KK LOOP FOR UNITS IN IOPA(K)
C
      300 CONTINUE
      400 CONTINUE
       IF(INT(5).EQ.1) WRITE(6,18)
   18 FORMAT("------EXIT ASSIGN")
      RETURN
      END
```

Subroutine: BASEDA

Function:

This routine inputs order and ship times and flying program data by base.

Description:

This routine is used by both the Events Generator and the Level Computation System to input order and ship time and flying program data by base. If the write flag IWT(12) equals 1, the routine prints out all input information.

```
*#RUN=;RIME/OBJ/BASEDA,O(BCD,NOGO)
*BASEDA,S
      SUBROUTINE BASEDA
      COMMON/NBASES/NBASES
      COMMON/OSTDLT/OSTDLT(8)
      COMMON/BFH/BFH(24,8)
      COMMON/IWT/IWT(20)
      CHARACTER IDENT*6
      CALL FPARAM(1,131)
C
C-----------------------------------------------------------
C
C            THIS ROUTINE INPUTS ORDER AND SHIP TIMES IN
C            ADDITION TO FLYING PROGRAMS BY BASE WHERE
C
C                 OSTDLT(K)=DEVIATIONS (IN DAYS) OF DEPOT
C                           ORDER AND SHIP TIME AT BASE K
C                           FROM THE AVERAGE ORDER AND SHIP
C                           TIME FOR THE GROUP.
C
C                  BFH(I,K)=FLYING HOURS AT BASE K DURING
C                           QUARTER I.
C
C-----------------------------------------------------------
C
C              INITIAL IDBUG.
C
C
      IDBUG=IWT(12)
     GTOTL=0.
      IF(IDBUG.EQ.1)WRITE(6,13)"*****ENTER BASEDA=--"
C-----------ADD A READ
C
          READ(5,13,END=500) NBASES ,(OSTDLT(K),K=1,NBASES)
   13 FORMAT(V)
C
C              REPORT SECTION.
C
      IF(IDBUG.EQ.0) GO TO 20
      WRITE(6,40)((K),I=1,NBASES)
   40 FORMAT(//T10,"BASE NUMBER"/
     &          /T15,8I10)
      WRITE(6,50)(OSTDLT(K),K=1,NBASES)
   50 FORMAT(/" OSTDLT ",T13,8(F8.0,2X))
   20 CONTINUE
      DO 100 I=1,24
      READ(5,13,END=500)  IDENT,(BFH(I,K),K=1,NBASES)
   60 FORMAT(  8X,A6,T13,8(F8.0,2X))
      IF(IDBUG.NE.1) GO TO 100
C
C          COMPUTE TOTAL PROGRAM OVER ALL BASES
C
```

```
      TOTAL=0.
      DO 70 J=1,NBASES
      TOTAL=TOTAL+BFM(I,J)
   70 CONTINUE
      GTOTL=GTOTL+TOTAL
      WRITE(6,73)TOTAL
   73 FORMAT(F95,"TOTAL =",F10.0)
      WRITE(6,60)  IDENT,(BFM(I,K),K=1,NBASES)
  100 CONTINUE
  500 CONTINUE
      IF(IDBUG.GE.1)WRITE(6,503)GTOTL
  503 FORMAT(//T90,"GRAND TOTAL=",F10.0//)
      RETURN
      END
```

Subroutine: EVTGN2, EVTGNB

Function:

This routine serves as the main program for the Events Generator. Program EVTGN2 generates exogenous events for LRU/SRU families that require no more than 2,000 events on the Future Events List (FEL) by any one time, while program EVTGNB is used for larger event families. At present, EVTGNB permits up to 3,000 events on the FEL at any given time.

Description:

For a detailed description of the Events Generator, see the Events Generator Chapter of Volume I.

```
*#RUN=:RTME/OBJ/EVTGN2.O(BCD,NOGO)
*EVTGN2.S
C
C-----------------------------------DEFINE GASP COMMONS-----
C
      DIMENSION NSET(1)
     COMMON QSET(500)
      EQUIVALENCE(NSET(1),QSET(1))
     COMMON /GCOM1/ ATRIB(30),JENNT,MFA,MFE(100),MLE(100),MSTOP,NCRDR,
&   NNAPO,NNAPT,NNATR,NNFIL,NNQ(100),NNTRY,NPRNT,PPARM(50,4),TNOV,TTBEG,
&     TTCLR,TTFIN,TTRIB(30),TTSET
     COMMON /GCOM6/ EENQ(100),IINN(100),KKRNK(100),MMAXQ(100),
&     QQTIM(100),SSOBV(25,5),SSTDV(25,6),VVNQ(100)
C
*-----------------------------------------------------------
     COMMON/IDBUG/IDBUG
     COMMON/ITIME/ITIME
     COMMON/NFGRP/NFGRP
     COMMON/NGROUP/NGROUP
     COMMON/NLGRP/NLGRP
     COMMON/NBASES/NBASES
     COMMON/NJOB/NJOB
     COMMON/NSRU/NSRU
     COMMON/NITEM/NITEM
     COMMON/IOUT/IOUT
     COMMON/IWT/IWT(20)
C
      COMMON/OSTDLT/OSTDLT(8)
      COMMON/BPH/BPH(24,8)
      COMMON/CPROB/CPROB(8)
C
     COMMON/NENTRY/NENTRY
     COMMON/NFEMAX/NFEMAX
     COMMON/NTIME/NTIME
      COMMON/ILOCFE/ILOCFE(2000)
      COMMON/JFSN/JFSN(2000)
      COMMON/JPOINT/JPOINT(2000)
      COMMON/JPRIOR/JPRIOR(2000)
      COMMON/JQTY/JQTY(2000)
      COMMON/JTIME/JTIME(2000)
      COMMON/JTYPE/JTYPE(2000)
C
     COMMON/IBRGN/IBRGN(16,40)
     COMMON/IRTS/IRTS(16,40)
     COMMON/IBCOND/IBCOND(16,40)
     COMMON/INRTS/INRTS(16,40)
     COMMON/IDCOND/IDCOND(16,40)
     COMMON/IDREP/IDREP(16,40)
     COMMON/IOVCND/IOVCND(16,40)
     COMMON/IPPROG/IPPROG(16,40)
     COMMON/IDRGN/IDRGN(16,40)
C
```

```
      COMMON/PSN/PSN(40)
      COMMON/UCOST/UCOST(40)
      COMMON/LTADM/LTADM(40)
      COMMON/LTPROD/LTPROD(40)
      COMMON/IBRT/IBRT(40)
      COMMON/IBDTT/IBDTT(40)
      COMMON/IDRT/IDRT(40)
      COMMON/IOST/IOST(40)
      COMMON/OIMDDR/OIMDDR(40)
      COMMON/RTSF/RTSF(40)
      COMMON/PDMF/PDMF(40)
      COMMON/EONF/EONF(40)
      COMMON/RMSTRF/RMSTRF(40)
      COMMON/DPDDR/DPDDR(40)
      COMMON/PJRCND/PJRCND(40)
      COMMON/PNJRC/PNJRC(40)
      COMMON/OIMBRR/OIMBRR(40)
      COMMON/BCONF/BCONF(40)
      COMMON/BJCNF/BJCNF(40)
      COMMON/BNCNF/BNCNF(40)
      COMMON/DONCNF/DONCNF(40)
      COMMON/RMSCNF/RMSCNF(40)
      COMMON/RMNRF/RMNRF(40)
      COMMON/RCOST/RCOST(40)
      COMMON/IBPD/IBPD(40)
      COMMON/IRINTR/IRINTR(40)
      COMMON/IOPA/IOPA(40)
C
      COMMON/NOINL/NOINL(40)
      COMMON/NBRTS/NBRTS(40)
      COMMON/NNRTS/NNRTS(40)
      COMMON/NBCOND/NBCOND(40)
      COMMON/NOVCNT/NOVCNT(40)
      COMMON/NODRPT/NODRPT(40)
      COMMON/NOSRUF/NOSRUF(40)
      COMMON/NDRGNT/NDRGNT(40)
      COMMON/NDCOND/NDCOND(40)
      COMMON/NDCONT/NDCONT(40)
      COMMON/NOVGNT/NOVGNT(40)
      COMMON/NDEXPO/NDEXPO(40)
C
      COMMON/LRTS/LRTS
      COMMON/LNRTS/LNRTS
      COMMON/LBRGN/LBRGN
      COMMON/LBCOND/LBCOND
      COMMON/LDRGN/LDRGN
      COMMON/LDRGNT/LDRGNT
      COMMON/LDREPT/LDREPT
      COMMON/LNRTST/LNRTST
      COMMON/LOVCNT/LOVCNT
      COMMON/LOVGNT/LOVGNT
      COMMON/LDCONT/LDCONT
```

```
      COMMON/LDCOND/LDCOND
C
      COMMON/INQTR/INQTR
      COMMON/ITDAY/ITDAY
      COMMON/ITMNTH/ITMNTH
      COMMON/ITQTR/ITQTR
      COMMON/ITYEAR/ITYEAR
      COMMON/ITINV/ITINV
C
      NAMELIST/DATA1/LNRTST,LDRGNT,LDCONT,LOVGNT,LOVCNT,LDREPT,NSBU,NITEM
C
      CHARACTER FSN*15
C
C
C           READ SIMULATION PARAMETERS
      WRITE(6,13)" NFGRP,  NLGRP, NBASES, INQTR, NREPL"
      READ(5,13) NFGRP,  NLGRP,KBASES,INQTR, NREPL
      WRITE(6,23) NFGRP,  NLGRP,KBASES,INQTR, NREPL
   23 FORMAT(10I7)
C
C
      READ(5,13)IWT
      WRITE(6,3)IWT
    3 FORMAT(//"WRITE FLAGS=",4(9X,5I2)//)
C
      IPRNT=IWT(8)
      IPRN2=IWT(9)
      IDBUG=IWT(10)
      IOUT=IWT(11)
C
C           INPUT BASE DATA
C
      CALL BASEDA
C
C
C           WRITE EXOGENOUS FILE HEADER RECORD
C
C           RESET NBASES TO CORRESPOND TO INPUT PARAMETER
C
C
      NBASES=KBASES
      IF(IOUT.EQ.1)WRITE(8)NFGRP,NLGRP,NBASES,INQTR,NREPL
C
C
      R=RANDU(-.1)
      ISEQ=
      KNTOUT=0
C
C
C           SET F.E.L.   LIMITS
C
      MAXFEL=0
```

```
      NFEMAX=2000
C
C
C        SET TIMING VARIABLES BASED ON 100 TIME UNITS PER DAY
C          7 DAYS/WEEK, 13 WEEKS/QUARTER, 4 QUARTERS/YEAR
C
      ITDAY=100
      ITWEEK=7*ITDAY
      ITMNTH=4*ITWEEK
      ITQTR=3*ITMNTH
      ITYEAR=4*ITQTR
C
C            SKIP THE FIRST (NFGRP-1) LRU GROUPS ON FILE 07
      CALL READF1
      NSKIP=NFGRP-1
      IF(NSKIP.LE.0) GO TO 60
      DO 50 NG=1,NSKIP
       NGROUP=NG
       CALL READFL(IENDF)
       IF(IENDF.EQ.1) GO TO 980
   50 CONTINUE
   60 CONTINUE
C
C
C            BEGIN GROUP LOOP
C
      DO 100 NG=NFGRP,NLGRP
       NGROUP=NG
      WRITE(6,13)"BEGIN LRU GROUP=",NG
   13 FORMAT(V)
C
C             SET LRU AND SRU HEADER DATA
C
      CALL READFL(IENDF)
      IF(IENDF.EQ.1)GO TO 980
C            BEGIN REPLICATION LOOP
C
      DO 990 KREPL=1,NREPL
       IF(IPRN2.EQ.1)WRITE(6,13)"BEGIN REPLICATION=",KREPL
C
C
C             RESET JOBS COUNTER
C
      NJOB=1000
C
C        INITIALIZE THE FUTURE EVENTS LIST
C
       CALL INFEL
C
C                 INITIALIZE DEPOT
C            EVENT COUNTERS
C
```

```
      NSRU=NITER=1
C
      CALL NOSET1
      IF(IPRNT.EQ.1) WRITE(6,DATA1)
C
C
C-----------------------------------------------------------
C           BEGIN QUARTER LOOP
C-----------------------------------------------------------
C
      DO 800 IQTR=1,INQTR
      ITINV=IQTR
C
C            INITIALIZE BASE PROBABILITY ARRAY CPROB.
C
      CALL RANBS1(ITINV)
C
C
C           INITIALIZE BASE EVENT COUNTERS
C
      CALL NOSET2(ITINV)
C           LOOP THROUGH EACH LRU FAILURE(L), WHERE L = 1,2,...,LBRGN
C
      IF(IPRNT.EQ.1) CALL SRUPRT
C
C
C-----------------BEGIN LRU BASE CONDEMNATION AND RTS GENERATION
C
      IF(LBRGN.GT.0)CALL LRUEVT(LBRGN,LRTS,NJOB,1)
C
C           GENERATE LRU NRTS EVENTS, IF ANY
C
      IF(LNRTS.GT.0)CALL LRUEVT(LNRTS,NLRUN,NJOB,2)
C
C           GENERATE LRU DEPOT REP GEN EVENTS, IF ANY
C
      NTOOV=LDRGN-LDCOND
      IF(LDRGN.GT.0)CALL LRUEVT(LDRGN,NTOOV,NJOB,3)
C
C=============END OF LRU EVENT GENERATION=============
C
  510 CONTINUE
      IF(IPRNT.NE.1) GO TO 580
     WRITE(6,DATA1)
      CALL SRUPRT
  580 CONTINUE
C
C           GENERATE INDEPENDENT SRU EVENTS, IF ANY
C
      CALL SRUIND
C
C
```

```
C          REMOVE ALL EVENTS FROM THE FEL, AND WRITE
C                THEM TO THE EXOGENOUS EVENT FILE.  AT END,
C                 GO TO 700
 60  CONTINUE
C
C          RECORD MAX ENTRIES TO THE F.E.L.
C
      MAXFEL=MAX(MAXFEL,NENTRY)
C
C          REMOVE EVENTS FROM THE F.E.L.
C
      JQEND=IQTR*ITQTR
      IF(NENTRY.LE.0) GO TO 700
      IF(NTIME.GT.JQEND) GO TO 700
       CALL REMOVE(KTIME,KTYPE,KFSN,KQTY,KPRI)
      IF(IPRN2.EQ.1)WRITE(6,617)KTIME,KTYPE,KFSN,KQTY,KPRI
  617 FORMAT("REMOVE*****",8I8)
      IF(IOUT.EQ.1)WRITE(8)KTIME,KTYPE,KFSN,KQTY,KPRI
      IF(IOUT.EQ.1)KNTOUT=KNTOUT+1
        GO TO 600
  700  CONTINUE
C
C          END OF QUARTER LOOP
C
      IF(IPRNT.EQ.1) WRITE(6,703)(IQTR,I=1,8)
  703 FORMAT( 8("-----END QTR",I3)/2(25("----")/))
C
 80  CONTINUE
C
C          SCHEDULE TYPE 10 EVENT...."END OF SIMULATED PERIOD"
C
      KTIME=99999999
      KTYPE=10
      KFSN=0
      KQTY=
      KPRI=
      IF(IPRN2.EQ.1)WRITE(6,617)KTIME,KTYPE,KFSN,KQTY,KPRI
     IF(IOUT.EQ.1)WRITE(8)KTIME,KTYPE,KFSN,KQTY,KPRI
      IF(IOUT.EQ.1)KNTOUT=KNTOUT+1
C
C          END OF REPLICATION LOOP
  990 CONTINUE
C
C          END OF GROUP LOOP
C
 1000 CONTINUE
C
C          END OF DATA SET GENERATION
C
      REWIND 08
      WRITE(6,13)"END OF EVENT SET GENERATION"
      WRITE(6,13)"MAX ENTRIES TO THE F.E.L =",MAXFEL
```

```
            WRITE(6,13)"NO. OF EVENTS WRITTEN TO EXOG FILE =",
     &          KNTOUT
     C
            STOP
       98   CONTINUE
     C
            REWIND 08
            WRITE(6,13)"READ END-OF-FILE ON 07.  STOP RUN."
            WRITE(6,13)"LAST LRU READ WAS=",FSN(1)
            STOP
            END
*W     7 MEMORY EXPANDED. USE $LIMITS OR CORE= OPTION FOR NEXT RUN
```

01 11-07-79 14.521 .S

```
*#RUN=;RIME/OBJ/EVTGNB,O(BCD,NOGO)
*EVTGN2.S
C
C-----------------------------------DEFINE GASP COMMONS-----
C
        DIMENSION NSET(1)
      COMMON QSET(500)
        EQUIVALENCE(NSET(1),QSET(1))
       COMMON /GCOM1/ ATRIB(30),JPVNT,MPA,MFE(100),MLE(100),MSTOP,NCRDR,
     &  NNAPO,NNAPT,NNATR,NNFIL,NNQ(100),NNTRY,NPRNT,PPARM(50,4),TNOW,TTBEG,
     &    TTCLR,TTFIN,TTRIB(30),TTSET
       COMMON /GCOM6/ EENQ(100),IINN(100),KKRNK(100),MMAXQ(100),
     &    QLTIM(100),SSOBV(25,5),SSTPV(25,6),VVNQ(100)
C
*-----------------------------------------------------------------
      COMMON/IDBUG/IDBUG
      COMMON/ITIME/ITIME
      COMMON/NFGRP/NFGRP
      COMMON/NGROUP/NGROUP
      COMMON/NLGRP/NLGRP
      COMMON/NBASES/NBASES
      COMMON/NJOB/NJOB
      COMMON/NSRU/NSRU
      COMMON/NITEM/NITEM
      COMMON/IOUT/IOUT
      COMMON/IWT/IWT(20)
C
       COMMON/OSTDLT/OSTDLT(8)
       COMMON/BFH/BFH(24,8)
       COMMON/CPROB/CPROB(8)
C
      COMMON/NENTRY/NENTRY
      COMMON/NFEMAX/NFEMAX
      COMMON/NTIME/NTIME
       COMMON/ILOCFE/ILOCFE(3000)
       COMMON/JFSN/JFSN(3000)
       COMMON/JPOINT/JPOINT(3000)
       COMMON/JPRIOR/JPRIOR(3000)
       COMMON/JQTY/JQTY(3000)
       COMMON/JTIME/JTIME(3000)
       COMMON/JTYPE/JTYPE(3000)
C
      COMMON/IBRGN/IBRGN(16,40)
      COMMON/IRTS/IRTS(16,40)
      COMMON/IBCOND/IBCOND(16,40)
      COMMON/INRTS/INRTS(16,40)
      COMMON/IDCOND/IDCOND(16,40)
      COMMON/IDREP/IDREP(16,40)
      COMMON/IOVCND/IOVCND(16,40)
      COMMON/IPPROG/IPPROG(16,40)
      COMMON/IDRGN/IDRGN(16,40)
C
```

```
      COMMON/FSN/FSN(40)
      COMMON/UCOST/UCOST(40)
      COMMON/LTADM/LTADM(40)
      COMMON/LTPROD/LTPROD(40)
      COMMON/IBRT/IBRT(40)
      COMMON/IBDTT/IBDTT(40)
      COMMON/IDRT/IDRT(40)
      COMMON/IOST/IOST(40)
      COMMON/OIMDDR/OIMDDR(40)
      COMMON/RTSF/RTSF(40)
      COMMON/PDMF/PDMF(40)
      COMMON/ROMF/ROMF(40)
      COMMON/RMSTRF/RMSTRF(40)
      COMMON/RPDDR/RPDDR(40)
      COMMON/PJRCND/PJRCND(40)
      COMMON/PMJRC/PMJRC(40)
      COMMON/OIMBRR/OIMBRR(40)
      COMMON/RCONF/RCONF(40)
      COMMON/RJCNF/RJCNF(40)
      COMMON/RMCNF/RMCNF(40)
      COMMON/DOMCNF/DOMCNF(40)
      COMMON/RMSCNF/RMSCNF(40)
      COMMON/RMNRF/RMNRF(40)
      COMMON/RCOST/RCOST(40)
      COMMON/IBPD/IBPD(40)
      COMMON/IRINTR/IRINTR(40)
      COMMON/IOPA/IOPA(40)
C
      COMMON/NOINL/NOINL(40)
      COMMON/NBRTS/NBRTS(40)
      COMMON/NNRTS/NNRTS(40)
      COMMON/NBCOND/NBCOND(40)
      COMMON/NOVCNT/NOVCNT(40)
      COMMON/NODRPT/NODRPT(40)
      COMMON/NOSRUP/NOSRUP(40)
       COMMON/NDRGNT/NDRGNT(40)
       COMMON/NDCOND/NDCOND(40)
       COMMON/NDCONT/NDCONT(40)
       COMMON/NOVGNT/NOVGNT(40)
       COMMON/NDEXPO/NDEXPO(40)
C
      COMMON/LRTS/LRTS
      COMMON/LNRTS/LNRTS
      COMMON/LBRGN/LBRGN
      COMMON/LBCOND/LBCOND
      COMMON/LDRGN/LDRGN
      COMMON/LDRGNT/LDRGNT
      COMMON/LDREPT/LDREPT
      COMMON/LNRTST/LNRTST
      COMMON/LOVCNT/LOVCNT
      COMMON/LOVGNT/LOVGNT
      COMMON/LDCONT/LDCONT
```

```
      COMMON/LDCOND/LDCOND
C
      COMMON/INOTR/INOTR
      COMMON/ITDAY/ITDAY
      COMMON/ITMNTH/ITMNTH
      COMMON/ITOTR/ITOTR
      COMMON/ITYEAR/ITYEAR
      COMMON/ITINV/ITINV
C
 NAMELIST/DATA1/INWTST,LDRGNT,LDCONT,LOVGNT,LCVCNT,LDRFPT,NSRU,NITFM
C
      CHARACTER FSN*15
C
C
C            READ SIMULATION PARAMETERS
      WRITE(6,13)" NFGRP, NLGRP, KBASES, INOTR, NREPL"
     READ(5,13) NFGRP, NLGRP,KBASES,INOTR, NREPL
      WRITE(6,23) NFGRP, NLGRP,KBASES,INOTR, NREPL
   23 FORMAT(10I7)
C
C
      READ(5,13)IWT
      WRITE(6,3)IWT
    3 FORMAT(/"WRITE FLAGS=",4(5X,5I2)//)
C
     IPRNT=IWT(8)
    IPRN2=IWT(9)
    IDBUG=IWT(10)
    IOUT=IWT(11)
C
C            INPUT BASE DATA
C
      CALL BASEDA
C
C
C            WRITE EXOGENOUS FILE HEADER RECORD
C
C            RESET NBASES TO CORRESPOND TO INPUT PARAMETER
C
C
      NBASES=KBASES
      IF(IOUT.EQ.1)WRITE(8)NFGRP,NLGRP,NBASES,INOTR,NREPL
C
C
      RFRANDU(-,1)
      ISEQ=0
      KNTOUT=0
C
C
C            SET F.E.L.   LIMITS
C
      MAXFEL=0
```

```
      NFEMAX=3000
C
C
C        SET TIMING VARIABLES BASED ON 100 TIME UNITS PER DAY
C          7 DAYS/WEEK, 13 WEEKS/QUARTER, 4 QUARTERS/YEAR
C
      ITDAY=100
      ITWEEK=7*ITDAY
      ITMNTH=4*ITWEEK
      ITQTR=3*ITMNTH
      ITYEAR=4*ITQTR
C
C          SKIP THE FIRST (NGRP-1) LRU GROUPS ON FILE 07
      CALL READF1
      NSKIP=NEGRP-1
      IF(NSKIP.LE.0) GO TO 60
      DO 50 NG=1,NSKIP
       NGROUP=NG
       CALL READFL(IENDF)
       IF(IENDF.EQ.1) GO TO 980
   50 CONTINUE
   60 CONTINUE
C
C
C          BEGIN GROUP LOOP
C
      DO 1000 NG=NFGRP,NLGRP
       NGROUP=NG
      WRITE(6,13)'BEGIN LRU GROUP=',NG
   13 FORMAT(V)
C
C           SET LRU AND SRU HEADER DATA
C
      CALL READFL(IENDF)
      IF(IENDF.EQ.1)GO TO 980
C          BEGIN REPLICATION LOOP
C
      DO 990 KREPL=1,NREPL
       IF(IPRN2.EQ.1)WRITE(6,13)'BEGIN REPLICATION=',KREPL
C
C
C           RESET JOBS COUNTER
C
      NJOB=1000
C
C        INITIALIZE THE FUTURE EVENTS LIST
C
       CALL INFEL
C
C                INITIALIZE DEPOT
C          EVENT COUNTERS
C
```

```
      NSNU=NITEM-1
C
      CALL NOSET1
      IF(IPRNT.EQ.1) WRITE(6,DATA1)
C
C
C-----------------------------------------------------
C           BEGIN QUARTER LOOP
C-----------------------------------------------------
C
      DO 800 IQTR=1,INQTR
      ITINV=IQTR
C
C           INITIALIZE BASE PROBABILITY ARRAY CPROB,
C
      CALL RANBS1(ITINV)
C
C
C           INITIALIZE BASE EVENT COUNTERS
C
      CALL NOSET2(ITINV)
C           LOOP THROUGH EACH LRU FAILURE(L), WHERE L = 1,2,...,LRBGN
C
      IF(IPRNT.EQ.1) CALL SNUPRT
C
C
C-------------------BEGIN LRU BASE CONDEMNATION AND RTS GENERATION
C
      IF(LRBGN.GT.0)CALL LRUEVT(LRBGN,LRTS,NJOB,1)
C
C           GENERATE LRU NRTS EVENTS, IF ANY
C
      IF(LNRTS.GT.0)CALL LRUEVT(LNRTS,NLRUN,NJOB,2)
C
C           GENERATE LRU DEPOT REP GEN EVENTS, IF ANY
C
      NTOOV=LBRGN-LDCOND
      IF(LDBGN.GT.0)CALL LRUEVT(LDBGN,NTOOV,NJOB,3)
C
C=========END OF LRU EVENT GENERATION=========
C
  510 CONTINUE
      IF(IPRNT.NE.1) GO TO 580
     WRITE(6,DATA1)
      CALL SNUPRT
  580 CONTINUE
C
C           GENERATE INDEPENDENT SRU EVENTS, IF ANY
C
      CALL SNUIND
C
C
```

```
T 01  11-07-79   14.521   .S
C            REMOVE ALL EVENTS FROM THE FEL, AND WRITE
C                 THEM TO THE EXOGENOUS EVENT FILE.  AT END,
C                  GO TO 700
  600 CONTINUE
C
C            RECORD MAX ENTRIES TO THE F.E.L.
C
      MAXFEL=MAX(MAXFEL,NENTRY)
C
C            REMOVE EVENTS FROM THE F.E.L.
C
      JQEND=IQTR*ITQTR
      IF(NENTRY.LE.0) GO TO 700
      IF(NTIME.GT.JQEND) GO TO 700
       CALL REMOVE(KTIME,KTYPE,KFSN,KQTY,KPRI)
      IF(IPRN2.EQ.1)WRITE(6,617)KTIME,KTYPE,KFSN,KQTY,KPRI
  617 FORMAT("REMOVE******",8I8)
      IF(IOUT.EQ.1)WRITE(8)KTIME,KTYPE,KFSN,KQTY,KPRI
      IF(IOUT.EQ.1)KNTOUT=KNTOUT+1
        GO TO 600
  700  CONTINUE
C
C            END OF QUARTER LOOP
C
      IF(IPRNT.EQ.1) WRITE(6,703)(IQTR,I=1,8)
  703 FORMAT( 8("-----END QTR",I3)/2(25("----")/))
C
  800 CONTINUE
C
C            SCHEDULE TYPE 10 EVENT...."END OF SIMULATED PERIOD"
C
      KTIME=99999999
      KTYPE=10
      KFSN=0
      KQTY=0
      KPRI=0
      IF(IPRN2.EQ.1)WRITE(6,617)KTIME,KTYPE,KFSN,KQTY,KPRI
     IF(IOUT.EQ.1)WRITE(8)KTIME,KTYPE,KFSN,KQTY,KPRI
      IF(IOUT.EQ.1)KNTOUT=KNTOUT+1
C
C            END OF REPLICATION LOOP
  990 CONTINUE
C
C            END OF GROUP LOOP
C
 1000 CONTINUE
C
C            END OF DATA SET GENERATION
C
      REWIND 08
      WRITE(6,13)"END OF EVENT SET GENERATION"
      WRITE(6,13)"MAX ENTRIES TO THE F.E.L =",MAXFEL
```

```
T 01   11-07-79    14.521    .S

            WRITE(6,13)"NO. OF EVENTS WRITTEN TO EXOG FILE =",
     &          ENTOUT
     C
            STOP
        980 CONTINUE
     C
            REWIND 08
            WRITE(6,13)"READ END-OF-FILE ON 07.  STOP RUN."
            WRITE(6,13)"LAST 'RU' READ WAS=",FSN(1)
            STOP
            END
*W     7 MEMORY EXPANDED. USE SLIMITS OR CORE= OPTION FOR NEXT RUN
```

Subroutine: LRUEVT

Function:

This routine controls the generation of all LRU exogenous events for the current quarter.

Calling Parameters:

NOLEFT	=	Total number of LRU reparable generations to be simulated for the current quarter.
LRTS	=	The number of these reparable generations that are to be repaired at the current location.
NJOB	=	The last job number assigned to a reparable generation.
MODEF	=	Mode of Failure Flag, where (1) indicates base condemnation or RTS events, (2) indicates a base NRTS event, and (3) indicates a depot rep gen event.

Description:

For a discussion of this routine, see the Events Generator Chapter of Volume I.

```
      *RUN*;RIME/OBJ/LRUEV2.O(BCD,NOGO)                                 26
      *LRUEV2,S   GENERATE LRU AND SRU EVENTS
      *******GENERATE EXOGENOUS EVENTS FOR THE CURRENT QUARTER
            SUBROUTINE LRUEVT(NOLEFT,LRTS,NJOB,MODEF)
      C               GENERATE LRU EVENTS, WHERE MODEF=
      C                     1  =  BASE CONDEMNATION AND RTS EVENTS
      C                     2  =  BASE NRTS EVENTS
      C                     3  =  DEPOT REP GEN EVENTS
      C
            COMMON/IWT/IWT(20)
            COMMON/NSRU/NSRU
            COMMON/IQPA/IQPA(1)
            COMMON/NOINL/NOINL(1)
            COMMON/NBRTS/NBRTS(1)
            COMMON/NNRTS/NNRTS(1)
            COMMON/NBCOND/NBCOND(1)
            COMMON/NOVCNT/NOVCNT(1)
            COMMON/NOSRUF/NOSRUF(1)
            COMMON/IDRT/IDRT(1)
            COMMON/IBDTT/IBDTT(1)
            COMMON/IBRT/IBRT(1)
            COMMON/LDCOND/LDCOND
            COMMON/LDCONT/LDCONT
            COMMON/LDRGN/LDRGN
            COMMON/LDRGNT/LDRGNT
            COMMON/NLRUF/NLRUR
            COMMON/LOVCNT/LOVCNT
            COMMON/LOVGNT/LOVGNT
            COMMON/LDREPT/LDREPT
            COMMON/LNRTST/LNRTST
      C
      C
            COMMON/ITOTR/ITOTR
             COMMON/ITDAY/ITDAY
             COMMON/NBASES/NBASES
             COMMON/ITINV/ITINV
            COMMON/ITIME/ITIME
            IF(IWT(4).EQ.1)WRITE(6,13)*******ENTER LRUEVT-*-NOLEFT=",
      &       NOLEFT,"   LRTS=",LRTS,"   NJOB=",NJOB,"   MODEF=",MODEF,
      &       "   LOVCNT",LOVCNT,"   LOVGNT",LOVGNT,"   LDREPT",
      &            LDREPT
       13 FORMAT(V)
      C
             IQTR=ITINV
      C
      C            START EVENTS GENERATION LOOP
            NUMB=NOLEFT
           DO 9020 LLL=1,NUMB
      C
      C            INCREMENT JOB COUNTER
      C
            NJOB=NJOB+1
```

```
C
C
C
      IF(IWT(4).EQ.1)WRITE(6,323)NOLEFT,LLL,NJOB,MODEF
  323 FORMAT(" NOLEFT=",I8," LRU=",I5," NJOB=",I5,
     &        " MODEF=",I5)
    5 CONTINUE
C
C
C           MONTE CARLO TO DETERMINE THE BASE NUMBER AND LRU STOCK
C           KEEPING UNIT ASSOCIATED WITH THIS FAILURE.
C
      CALL RANBAS(NB)
C
      IF(MODEF.EQ.1)NSKLRU=NB+1
      IF(MODEF.EQ.3) NSKLRU=NBASES+2
      IF(MODEF.EQ.2)NSKLRU=NB+1
      IF(IWT(4).EQ.1)WRITE(6,13)"   NB=",NB,"   NSKLRU=",NSKLRU
C
C           COMPUTE FAILURE TIME FOR THIS LRU,ASSUMING
C           FAILURE EVENTS ARE UNIFORMLY DISTRIBUTED THROUGHOUT
C           A QUARTER.
C
      ITT=RANDU(.2)*ITQTR
      ITIME=(IQTR-1)*ITQTR + ITT
C
C           SCHEDULE THE FAILURE EVENT FOR THIS LRU
C
      JTIME=ITIME
      CALL ENTER(JTIME,14,NSKLRU,1,NJOB)
C
C
C           SCHEDULE A REQUISITION TO REPLACE THE FAILED LRU
C
      JTIME=ITIME + 10
      CALL ENTER(JTIME,1,NSKLRU,1,1)
C
C           BRANCH BASED ON FAILURE MODE(MODEF).
C
      GO TO (100,700,600),MODEF
C
C            RTS EVENT
C
C
  100 CONTINUE
      CALL ASSIGN(NOLEFT,NOSRUF,NOINL)
      IF(IWT(4).EQ.1)WRITE(6,327) (NOINL(KKK),KKK=1,NSRU)
  327 FORMAT(T50,"NOINL(K)=",40I2)
C           AT THIS POINT, THE VARIABLE NOINL(K) EQUALS THE NUMBER
C           OF BASE REPARABLE GENERATIONS THAT ARE TO BE ASSOCIATED
C           WITH THE CURRENT LRU REPARABLE GENERATION. THE FOLLOWING
C           LOGIC FURTHER REFINES THE FAILURE TYPES, AND SCHEDULES
```

```
      C          ALL RELATED REQUISITION, REPAIR, AND CONDEMNATION ACTIONS.
      C
      C
      C          FAILURE REMAINS AT THIS LOCATION
      C
      C
      C           MONTE CARLO TO DETERMINE IF THE LRU IS REPARABLE.
      C          IF NOT, GO TO 300.
      C
            LRUREP=1
            R=RANDU(.2)
            PRTS=FLOAT(LRTS)/FLOAT(NOLEFT)
            IF(R.GT.PRTS)LRUREP=0
      C
            IF(INT(A).EQ.1)WRITE(6,13)"LRTS=",LRTS,"  NOLEFT=",NOLEFT,
      8           "  PRTS=",PRTS,"   LRUREP=",LRUREP
            IF(INT(A).EQ.1) WRITE(6,14)
         14 FORMAT('      ........NOTE   LRUREP=0 MEANS LRU IS CONDEMNED.')
      C
            IF(R.LE.PRTS)LRTS=LRTS-1
      C
      C          COMPUTE THE TOTAL PARTS (NNEED) NEEDED TO REPAIR THIS LRU.
      C
            NNEED=0
            DO 110 K=1,NSRU
            NNEED=NNEED+NOINL(K)
        110 CONTINUE
      C
      C          IF LRU IS TO BE CONDEMNED, GO TO 300.
            IF(LRUREP.LE.0) GO TO 300
      C
      C
      C     IF NO PARTS ARE NEEDED (IE. IF NNEED=0), SCHEDULE AN  LRU
      C          REPAIR COMPLETION EVENT FOR STOCKKEEPING NUMBER NSKLRU
      C
            IF(NNEED.GE.1) GO TO 200
            JTIME=ITIME+IBRT(NSKLRU)*ITDAY
            CALL ENTER(JTIME,18,NSKLRU,1,NJOB)
      C
      C          COMPLETE LRU EVENT SCHEDULING.
      C
            GO TO 9000
      C
      C           THERE HAS BEEN AN SRU FAILURE.  THE FOLLOWING LOGIC
      C              SCHEDULES ALL RELATED RES, NRTS, CONDEMNATION,
      C                AND REQUISITION AND REPAIR COMPLETION EVENTS.
      C
      C
        200 CONTINUE
      C
      C          SCHEDULE "A BEGIN WAIT" EVENT FOR THE LRU
      C
```

```
      JTIME=ITIME + 5
      CALL ENTER(JTIME,16,NSKLRU,NNEED,NJOB)
C
C
C           SCHEDULE ALL RELATED SRU EVENTS
C
      CALL SRUEVT(LRUREP,NSKLRU)
      GO TO 9000
C
C           LRU IS TO BE CONDEMNED
C
  300 CONTINUE
C
C           SCHEDULE LRU CONDEMNATION EVENT(EVENT CODE 15)
C
      JTIME=ITIME+10
      CALL ENTER(JTIME,15,NSKLRU,1,NJOB)
C
C           SCHEDULE ALL RELATED SRU EVENTS
C
      CALL SRUEVT(LRUREP,NSKLRU)
       GO TO 9000
C
C
C=========================LRU DEPOT REP GEN
C
  600 CONTINUE
C
C           MONTE CARLO TO DETERMINE IF REP GEN IS
C              A DEPOT CONDEMNATION
C              IF NOT, GO TO 720 AND WRTS THIS LRU TO THE DEPOT
C
C
      DCONR=FLOAT(LDCOND)/FLOAT(NOLEFT)
      R=RANDU(.2)
C
      IF(IWT(4).EQ.1)WRITE(6,13)"<<<<<<DEPOT REP GEN",
     &        "   LDRGNT=",LDRGNT,"   LDRGN=",LDRGN,
     &        "   LDCOND=",LDCOND,"   DCONR=",DCONR,
     &        "   R=",R
C
C           UPDATE TOTAL COUNTS FOR DEPOT REP GENS
C
      LDRGNT=LDRGNT-1
C
      IF(R.GE.DCONR) GO TO 710
C
C            SCHEDULE A DEPOT CONDEMNATION EVENT
C
      LDCOND=LDCOND-1
      LDCONT=LDCONT-1
      JTIME=ITIME+20
```

```
      CALL ENTER(JTIME,15,NSELRU,1,NJOB)
C
      GO TO 9000
C
C
C=================================LRU NRTS EVENTS
C
      700 CONTINUE
C
C           SCHEDULE NRTS EVENT
C
      LNRTST=LNRTST-1
  710 CONTINUE
      JTIME=ITIME+30
      CALL ENTER(JTIME,19,NSELRU,1,NJOB)
C
C           MONTE CARLO TO DETERMINE IF LRU GENERATION IS
C              OVERHAUL CONDEMNED
C
  720 CONTINUE
      DCONR=FLOAT(LOVCNT)/FLOAT(LDREPT+LOVCNT)
      R=RANDU(.2)
C
      IF(IWT(4).EQ.1)WRITE(6,13)"<<<<<<<NRTS EVENT",
     &      "   LOVCNT=",LOVCNT,"   LDREPT=",LDREPT,
     &      "   DCONR=",DCONR,"   R=",R
C
      IF(DCONR.LE.R) GO TO 750
C
C           SCHEDULE DEPOT OVERHAUL CONDEMNATION EVENT.
C
      LOVCNT=LOVCNT-1
      LOVGNT=LOVGNT-1
      JTIME=ITIME+IBDTT(1)*ITDAY
      CALL ENTER(JTIME,18,1,1,NJOB)
      GO TO 9000
C
C           SCHEDULE DEPOT REPAIR COMPLETION
C
  750 CONTINUE
      LDREPT=LDREPT-1
      LOVGNT=LOVGNT-1
      JTIME=ITIME+(IBDTT(1)+IDRT(1))*ITDAY
      CALL ENTER(JTIME,18,1,1,NJOB)
      GO TO 9000
C
C
C=================================END OF EVENTS LOOP
 9000 CONTINUE
C
C              DECREASE NUMBER OF LRU'S REMAINING.
C
```

Subroutine: NOSET1, NOSET2

Function:

This routine initializes counter arrays that determine the number of LRU and SRU events to be generated. Entry point NOSET1 initializes depot event counters, while NOSET2(KQ) initializes counters for quarter KQ.

Description:

In this routine, the variable N denotes an item number. The LRU is denoted by N=1, while item numbers of N=2, N=3,... correspond to the first SRU, the second SRU, and so on. For definitions of variables used in this routine, see the Events Generator Chapter of Volume I.

Entry point NOSET1 initializes depot events counters. The routine first computes the total number of LRU depot events associated with the variables LDCONT, LOVCNT, LNRTST, and LDRGNT by totalling corresponding input variables over all 16 quarters in the planning horizon. Next, the routine checks if the total number of depot condemnations (LDCONT) exceeds the total number of depot reparable generations (LDRGNT). If so, LDCONT is reset to LDRGNT. Next, similar calculations are performed for each SRU. Finally, the routine checks if the total number of depot level SRU failure exposures exceed the number of SRU depot reparable generations. If so, the excess failures are assigned to a randomly selected quarter and program control returns to the calling program.

Entry point NOSET2(KQ) initializes counters used to simulate reparable generations for quarter KQ. The routine first sets the LRU counter variables LBCOND, LRTS, LDRGN, LNRTS, LBRGN, and LDCOND to the corresponding input variables associated with quarter KQ. Next, the routine similarly initializes the SRU counter variables NOSRUF, NBRTS, NNRTS, NBCOND, NDRGN, and NDCOND for each SRU. Program control then returns to the calling routine.

```
*SRUN=;RIME/OBJ/NOSET1,O(BCD,NOGO)
*NOSET1,S
          SUBROUTINE NOSET1
C
C          THIS ROUTINE INITIALIZES COUNTER ARRAYS THAT DETERMINE
C           THE NUMBER OF LRU AND SRU EVENTS TO BE GENERATED.
C           ENTRY POINT NOSET1 INITIALIZES DEPOT EVENTS COUNTERS, WHILE
C          NOSET2(KQ) INITIALIZES COUNTERS FOR QUARTER KQ.
C     NOTE--THE LRU IS ITEM NUMBER N=1, ITEMS N=2,3, AND GREATER CORRESP
C                TO THE FIRST SRU, THE SECOND SRU, ETC.
      COMMON/NENTRY/NENTRY
      COMMON/IDBUG/IDBUG
      COMMON/NBASES/NBASES
      COMMON/NSRU/NSRU
      COMMON/LBCOND/LBCOND
      COMMON/LRTS/LRTS
      COMMON/LNRTS/LNRTS
      COMMON/LNRTST/LNRTST
      COMMON/LBRGN/LBRGN
       COMMON/LDRGN/LDRGN
      COMMON/LDRGNT/LDRGNT
       COMMON/LDCOND/LDCOND
      COMMON/LOVGNT/LOVGNT
      COMMON/LDCONT/LDCONT
      COMMON/LOVCNT/LOVCNT
      COMMON/LDREPT/LDREPT
      COMMON/IOPA/IOPA(40)
      COMMON/NOINL/NOINL(40)
      COMMON/NBRTS/NBRTS(40)
      COMMON/NNRTS/NNRTS(40)
      COMMON/NNRTST/NNRTST(40)
      COMMON/NBCOND/NBCOND(40)
      COMMON/NOVCNT/NOVCNT(40)
      COMMON/NODRPT/NODRPT(40)
      COMMON/NOSRUF/NOSRUF(40)
      COMMON/NDRGN/NDRGN(40)
       COMMON/NDRGNT/NDRGNT(40)
       COMMON/NDCONT/NDCONT(40)
       COMMON/NDCOND/NDCOND(40)
       COMMON/NOVGNT/NOVGNT(40)
       COMMON/NDEXPO/NDEXPO(40)
      COMMON/NINDGN/NINDGN(16,40)
       COMMON/NOINDG/NOINDG(40)
C
       COMMON/IBCOND/IBCOND(16,40)
       COMMON/IRTS/IRTS(16,40)
       COMMON/IDCOND/IDCOND(16,40)
       COMMON/INRTS/INRTS(16,40)
       COMMON/IDRGN/IDRGN(16,40)
      COMMON/IOVCND/IOVCND(16,40)
C
      COMMON/INQTR/INQTR
```

NOSET1

```
      COMMON/ITDAY/ITDAY
      COMMON/ITOTB/ITOTB                                                54
      COMMON/ITINV/ITINV
      COMMON/NTIME/NTIME
      COMMON/ITIME/ITIME
      COMMON/IWT/IWT(20)
C
      IF(IWT(3).EQ.1)WRITE(6,383)"****NOBET1 CALLED"
C
C                 ASSIGN THESE FAILURES TO A RANDOMLY SELECTED QUARTER
C
C     COMPUTE TOTALS FOR THE PLANNING HORIZON(QUARTERS K=1,2,....,IQPH):
C                                 (1)DEPOT LRU REPARABLE GENERATIONS
      IQPH=16
C
          LDCONT=0
         LOVCNT=0
           LOVGNT= 0
      LNRTST=0
      LDRGNT=0
      DO 100 KKK=1,IQPH
      K=KKK
         LDCONT=LDCONT+IDCOND(K,1)
         LOVCNT=LOVCNT+IOVCND(K,1)
      LNRTST=LNRTST+INRTS(K,1)
      LDRGNT=LDRGNT+IDRGN(K,1)
  100 CONTINUE
C
C           LIMIT CONDEMNATIONS TO NO MORE THAN TOTAL GENERATIONS
C
      IF(LDCONT.GT.LDRGNT)LDCONT=LDRGNT
      LOVGNT=LNRTST+LDRGNT-LDCONT
      IF(LOVCNT.GT.LOVGNT)LOVCNT=LOVGNT
      LDREPT=LOVGNT-LOVCNT
C
C                                 (2)SRU DEPOT GENERATIONS
C
      DO 400 JJJ=1,NSRU
      J=JJJ
          JJ=J+1
          NDRGNT(J)=0
           NDCONT(J)=0
      NNRTST(J)=0
          NOVGNT(J)=0
        NOVCNT(J)=0
      DO 200 KKK=1,IQPH
       KQ=KKK
      NNRTST(J)=NNRTST(J)+INRTS(KQ,JJ)
          NDRGNT(J)=NDRGNT(J)+IDRGN(KQ,JJ)
C
C           SRU DEPOT CONDEMNATIONS
C
```

```
                NDCONT(J)=NDCONT(J)+IDCOND(KO,JJ)
                NOVCNT(J)=NOVCNT(J)+IOVCND(KO,JJ)
      C
      C
      C
       200 CONTINUE
      C-----------------END OF QUARTER KO LOOP
      C
      C           LIMIT CONDEMNATIONS
      C
                IF(NDCONT(J).GT.NDRGNT(J))NDCONT(J)=NDRGNT(J)
      C
          NOVGNT(J)=NNRTST(J)+NDRGNT(J)-NDCONT(J)
           IF(NOVCNT(J).GT.NOVGNT(J))NOVCNT(J)=NOVGNT(J)
      C
      C
      C           COMPUTE NO OF SRU DEPOT REPAIRS
      C
           NODRPT(J)=NOVGNT(J)-NOVCNT(J)
      C
      C
      C           SET TOTAL SRU DEPOT REPAIRS OVER HORIZON
      C
           NODRPT(J)=NOVGNT(J)-NOVCNT(J)
      C
      C                          (5)SRU FAILURE EXPOSURES

              NDEXPO(J)=(LOVGNT+LDCONT)*IOPA(J)
      C
      C                          (6)SRU INDEPENDENT FAILURES
      C
           NOIND=0
           IF(NDEXPO(J).LT.NDRGNT(J))NOIND=NDRGNT(J)-NDEXPO(J)
      C
      C           ASSIGN THESE NOIND FAILURES TO A RANDOMLY SELECTED QUARTER.
      C
            DO 300 KKK=1,IOPH
             KQ=KKK
        300 NINDGN(KQ,J)=0
           IF(NOIND.LE.0)GO TO 320
      C
           DO 310 NF=1,NOIND
            KQ=RANDU(.2)*IOPH + 1
            NINDGN(KQ,J)=NINDGN(KQ,J)+1
        310 CONTINUE
      C
       320 CONTINUE
      C
       321 CONTINUE
           IF(IWT(3).NE.1) GO TO 360
             WRITE(6,323)J,(NINDGN(KQ,J),KQ=1,16)
        323  FORMAT("--IND. DEP REP GENS FOR SRU ",I4," FOLLOW"/
```

```
     &              16I4)
      C
      C
      C                    SET DEPOT COUNTERS
      C
  360 CONTINUE
        IF(NODRPT(J).GE.0) GO TO 381
            WRITE(6,380)J,NODRPT(J),ITIME
  380 FORMAT(///T10,'*** WARNING  ***'/
     &        T5,'NODRPT(',I2,')= ',I6,' WHICH IS RESET TO 0')
        NODRPT(J)=0
  381 CONTINUE
        IF(IWT(3).NE.1) GO TO 397
        WRITE(6,383)"  J   LOVGNT NDRGNT NDCONT NDEXPO IOPA",
     &        "  NOVCNT  NODRPT NOVGNT"
  383 FORMAT(V)
        WRITE(6,393) J,LOVGNT,NDRGNT(J),NDCONT(J),NDEXPO(J),IOPA(J),
     &          NOVCNT(J),NODRPT(J),NOVGNT(J)
  393 FORMAT(I3,I9,8I7)
  397 CONTINUE
      C
        400 CONTINUE
      C
      C-----------------------------END OF SRU LOOP
      C-----------------------------END OF NOSET1
        IF(IWT(3).EQ.1)WRITE(6,383)"-----EXIT NOSET1"
      C
                RETURN
      C
      C                SET UP ARRAYS FOR QUARTER KQ
      C
            ENTRY NOSET2(KQ)
        IF(IWT(3).EQ.1)WRITE(6,383)"***ENTER NOSET2"
      C     GENERATE BASE LEVEL LRU EVENTS, AND ALL RELATED SRU EVENTS
      C                          COMPUTE APPLICABLE BASE GENERATIONS
      C
        LBCOND=IBCOND(KQ,1)
          LRTS=IRTS(KQ,1)
        LDRGN=IDRGN(KQ,1)
        LNRTS=INRTS(KQ,1)
        LBRGN=LRTS+LBCOND
        LDCOND=IDCOND(KQ,1)
        IF(LDCOND.GT.LDRGN)LDCOND=LDRGN
        IF(IWT(3).EQ.1)WRITE(6,383)"LBRGN=",LBRGN,"   LRTS=",LRTS,
     &          "   LBCOND=",LBCOND,"   LNRTS=",LNRTS,
     &        "   LDRGN=",LDRGN,"   LDCOND=",LDCOND
      C
      C                          FOR EACH SRU(K),K=1,2,...,NSRU
        IF(IWT(3).EQ.1)
     &   WRITE(6,383)"  K   NOSRUF  NRRTS NNRTS  NBCOND NOINDG",
     &   " NDCOND NDRGN"
      C
```

```
      DO 500 K=1,NSRU
      KK=K+1
C
C       SET SRU BASE GENERATION DATA
C
      NOSBUF(K)=IRTS(KQ,KK)+INRTS(KQ,KK)+IBCOND(KQ,KK)
      NBRTS(K)=IRTS(KQ,KK)
      NNRTS(K)=INRTS(KQ,KK)
      NBCOND(K)=IBCOND(KQ,KK)
C
C       SET SRU DEPOT GENERATION DATA
C
C       SET SRU DEPOT REP GENS ASSUMING THESE
C          GENERATIONS ARE INDEPENDENT OF LRU REP GENS.
C
    NDRGN(K)=IDRGN(KQ,KK)
C
C
    NDCOND(K)=IDCOND(KQ,KK)
    IF(NDCOND(K).GT.NDRGN(K))NDCOND(K)=NDRGN(K)
C
    IF(IWT(3).NE.1) GO TO 487
    WRITE(6,493)K,NOSBUF(K),NBRTS(K),NNRTS(K),NBCOND(K),
   &     NOINDG(K),NDCOND(K),NDRGN(K)
  493 FORMAT(I3,2X,8I7)
C
C       SET THE NUMBER OF LRU DEPOT REP GENS FOR THIS QUARTER,
C        AND THE NUMBER OF LRU DEPOT CONDEMNATIONS.
C
  487 CONTINUE
  500 CONTINUE
C
    IF(IWT(3).EQ.1)WRITE(6,383)"-->EXIT NOSET2"
      RETURN
      END
```

Subroutine: RANBAS, RANBS1

Function:

These routines uses Monte Carlo techniques to determine a randomly selected base.

Description:

Entry point RANBS1(IQTR) computes the cumulative probability of a failure at a given base K during quarter IQTR, and records the corresponding probabilities in the COMMON variable CPROB(K). This routine is called at the beginning of each simulated quarter. When a randomly selected base is needed, entry point RANBAS(NB) is called. This routine first obtains a uniform random number R in the range 0 to 1 by calling subroutine RANDU. It then does a table look-up using the base probability array CPROB to determine the base NB corresponding to the random number R. Program control then returns to the calling program.

```
      SUBROUTINE RANBAS(NB)
C
C       MONTE CARLO TO DETERMINE REP GEN BASE LOCATION NB, WHERE
C          CPROB(KB) DENOTES THE CUMULATIONVE PROBABILITY OF
C           A FAILURE AT BASE KB.
C          CPROB(K) MUST BE COMPUTED BY CALLING RANBS1 PRIOR TO
C           CALLING THIS ROUTINE.
C
      COMMON/IWT/IWT(20)
      COMMON/BFH   /BFH(24,8)
      COMMON/CPROB/CPROB(8)
      COMMON/NBASES/NBASES
      R = RANDU(2)
      DO 20 KB = 1,NBASES
   20 IF(R.LE.CPROB(KB)) GO TO 30
   30 NB = KB
      RETURN
C
      ENTRY RANBS1(IQTR)
C          THIS ROUTINE COMPUTES THE CUMULATIVE PROBABILITY
C             OF A FAILURE AT A GIVEN BASE K.
C
C
      IDBUG=IWT(13)
C
C     COMPUTE TOTAL BASE FLYING HOURS THIS QUARTER
C
      TBFH = 0.
      DO 410 KB = 1, NBASES
      TBFH = TBFH + BFH(IQTR,KB)
  410 CONTINUE
C
C       COMPUTE CUMULATIVE PROBOBILITY OF A REP GEN AT BASE KB
C
      TOTL = 0.
      DO 420 KB = 1 , NBASES
      TOTL = TOTL + BFH(IQTR,KB)
      CPROB(KB)=TOTL/TBFH
  420 CONTINUE
C
C          IF IDBUG.EQ.1, PRINT CPROB ARRAY.
C
      IF(IDBUG.EQ.1)WRITE(6,423)NBASES,TBFH,
     &   (K,BFH(IQTR,K),CPROB(K),K=1,NBASES)
  423 FORMAT(//"-----RANBS1--  NBASES=",I3,
     &          "   TBFH=",F10.0,
     &          8(/T40,"BASE=",I3,"   BFH=",F8.0,
     &             "   CPROB=",F8.3))
C
      RETURN
      END
```

RANBAS

Subroutine: READFL, READF1

Function:

This routine inputs D041 data describing the characteristics of individual LRU and SRUs. The routine is used by both the Events Generator and the Levels Computation System.

Description:

Entry point READF1 is called first to rewind Item Data file 07, and to read the first record on this file. Entry point READFL (IENDF) is then called to interpret the previously read record. The parameter IENDF is first set equal to 0. If an End-of- File is encountered during subsequent read operations, IENDF is set to 1 and program control returns to the calling program.

In READFL, the previously read data record is decoded to determine the record type. Logic then branches to the appropriate logic for decoding that specific type of input record. After decoding the record and setting the values of the corresponding COMMON variables, the routine reads the next record from file 07, and decodes it. If the record type of newly read record is different from that of previous record, logic jumps to statement 2000; otherwise logic again jumps to the decode logic described above.

When a new record is encountered, the routine checks (in lines 2120-2240) if the input data lies within "reasonable" ranges. If not. the associated variables are reset to standard values. Next, the routine uses the input data to set major time

variables used in the RIME model. These time variables include the base repair time, depot repair times, and various transportation times.

If the flag IOUT equals 1, subroutine READFL outputs an item identification record to the Exogenous Event File (File 08). This option is not used when READFL is called by the Levels Computation System.

The above operations are continued until all input records for the current LRU/SRU family have been processed. After all records have been processed, if IOUT equals 1, the trailer record of zeros is written to File 08. Finally, logic returns to the calling program.

```
*#RUN=;RIME/OBJ/READF2,O(BCD,NOGO)
*READF2,S
      SUBROUTINE READFL(IENDF)
C-----------------------BASEDA,S DATA
      COMMON/NBASES/NBASES
      COMMON/OSTDLT/OSTDLT(8)
C-----------------------
      COMMON/IOUT/IOUT
      COMMON/NGROUP/NGROUP
      COMMON/NFGRP /NFGRP
      COMMON/IWT/IWT(1)
      COMMON/NITEM/NITEM
C
      COMMON/IBRGN/IBRGN(16,40)
      COMMON/IRTS/IRTS(16,40)
      COMMON/IBCOND/IBCOND(16,40)
      COMMON/INRTS/INRTS(16,40)
      COMMON/IDCOND/IDCOND(16,40)
      COMMON/IDREP/IDREP(16,40)
      COMMON/IOVCND/IOVCND(16,40)
      COMMON/IPPROG/IPPROG(16,40)
      COMMON/IDRGN/IDRGN(16,40)
      COMMON/FSN/FSN(40)
      COMMON/UCOST/UCOST(40)
      COMMON/LTADM/LTADM(40)
      COMMON/LTPROD/LTPROD(40)
      COMMON/IBRT/IBRT(40)
      COMMON/IBDTT/IBDTT(40)
      COMMON/IDORT/IDORT(40)
      COMMON/IDRT/IDRT(40)
      COMMON/IOST/IOST(40)
      COMMON/OIMDDR/OIMDDR(40)
      COMMON/RTSF/RTSF(40)
      COMMON/PDMF/PDMF(40)
      COMMON/EOHF/EOHF(40)
      COMMON/RMSTRF/RMSTRF(40)
      COMMON/DPDDR/DPDDR(40)
      COMMON/PJRCND/PJRCND(40)
      COMMON/PNJRC/PNJRC(40)
      COMMON/OIMBRR/OIMBRR(40)
      COMMON/BCONF/BCONF(40)
      COMMON/EJCNF/EJCNF(40)
      COMMON/ENCNF/ENCNF(40)
      COMMON/DOHCNF/DOHCNF(40)
      COMMON/RMSCNF/RMSCNF(40)
      COMMON/RMNRF/RMNRF(40)
      COMMON/RCOST/RCOST(40)
      COMMON/IBPD/IBPD(40)
      COMMON/IRINTR/IRINTR(40)
      COMMON/IQPA/IQPA(40)
      COMMON/SIMDDR/SIMDDR(40)
      COMMON/IC1/IC1
```

```
      COMMON/IC2/IC2
C-----------------------------
      DIMENSION IDSHIP(8)
      DIMENSION IZERO(17)
       DATA IZERO/17*0/
C-----------------------------
      CHARACTER ISNREC*100,INREC*182,FSN*15,NAME*10,ERRC*1
      CHARACTER WFSN*15
      CHARACTER ALC*2
C
      IENDF=0
      IODH=16
      NITEM=0
      N=1
C
      WRITE(6,13)NGROUP,NFGRP,IWT(1),IWT(2)
   13 FORMAT( " --NGROUP=",I3,"  NFGRP=",I3,"  IWT1=",
     &          I3,"   IWT2=",I3)
C
C
      IPRINT=0
      IF(NGROUP.GE.NFGRP)IPRINT=IWT(1)
  100 CONTINUE
C
C
C                WHAT TYPE OF RECORD HAVE WE READ?
C
       GO TO(110,120,130,140,150,160,170,180,190,200,210),IREC
C
C                TYPE#1 RECORD.
C
      110 DECODE(INREC,113)ALC,FSN(N),NAME,UCCST(N),ERRC,
     &      UI,LTADM(N),LTPROD(N),IBRT(N),IDRCD,BPCODE,SMCODE,
     &       MPCODE,IOST(N),IOIMSL,NJRSL,IDFSL,CINDDR(N),SIMDDR(N),
     &       RTSF(N),PDMF(N),EOHF(N),RMSTRF(N),CPBDR(N),BNRTSF,
     &       PJRCND(N),PNJRC(N),OIMBRR(N),BCONF(N),EJCNF(N),FNCNF(N),
     &       DOHCNF(N),RMSCNF(N),RMNRF(N),NAPPL,NUSERS,IENDAT,
     &       IESSN,RCOST(N),IEPHRS,JRSLD,IBPD(N),IRINTR(N),ISORM,
     &       ISFLOW,ISRTIN,IBLT,IPBD,IOPA(N),IAP
      113 FORMAT(A2,A15,A10,F9.2,A1,A2,I1,I2,2I3,A2,2A4,
*W 1282 INCOMPATIBLE W.D FIELD IN 'F' SPECIFICATION
*W 1282 INCOMPATIBLE W.D FIELD IN 'F' SPECIFICATION
*W 1282 INCOMPATIBLE W.D FIELD IN 'F' SPECIFICATION
*W 1282 INCOMPATIBLE W.D FIELD IN 'F' SPECIFICATION
*W 1282 INCOMPATIBLE W.D FIELD IN 'F' SPECIFICATION
*W 1282 INCOMPATIBLE W.D FIELD IN 'F' SPECIFICATION
*W 1282 INCOMPATIBLE W.D FIELD IN 'F' SPECIFICATION
     &       3I2,I3,F5.4,A1,4F3.2,F5.4,F3.2,2F3.2,F5.4,6F3.2,I5,I3,
*W 1222 THE CHARACTERS PRECEDING ABOVE FORMAT ERROR ARE ,F5.4,
*W 1222 THE CHARACTERS PRECEDING ABOVE FORMAT ERROR ARE 4F3.2,
*W 1222 THE CHARACTERS PRECEDING ABOVE FORMAT ERROR ARE ,F5.4,
*W 1222 THE CHARACTERS PRECEDING ABOVE FORMAT ERROR ARE ,F3.2,
```

```
**W  1222 THE CHARACTERS PRECEDING ABOVE FORMAT ERROR ARE 2F3.2.
**W  1222 THE CHARACTERS PRECEDING ABOVE FORMAT ERROR ARE ,F5.4.
**W  1222 THE CHARACTERS PRECEDING ABOVE FORMAT ERROR ARE 6F3.2.
     &          I4,A3,F9.2,I5,I2,6I3,I4,I4,I3)
          JC1=IC1
          JC2=IC2
          JC3=IC3
          JSEQ=ISEQ
          IF(IPRINT.LT.1) GO TO 1000
        WRITE(6,117) ISEQ,IC1,IC2,IREC,ALC,FSN(N),NAME,UCOST(N),ERRC,NGROUP
        117 FORMAT('  SEQ# IC1 IC2 IREC ALC        FSN          ',
     &    '  NOUN        COST      ERRC   GRP',/I6,3I4,3X,A2,2X,A15,
     &              A10,F10.2,7X,A1,I6//)
              GO TO 1000
     C
     C              TYPE #2 RECORD
     C
           120 CONTINUE
     C
     C              TYPE #3 RECORD
     C
           130 DECODE(INREC,133) (IBRGN(K,N),K=1,IQPN)
           133 FORMAT(2X,16I6)
          IF(IPRINT.GE.2)CALL PRTPAS(IREC,N,IBRGN(1,N))
              GO TO 1000
     C
     C              TYPE #4 RECORD
     C
           140 DECODE(INREC,133) (IRTS(K,N),K=1,IQPN)
          IF(IPRINT.GE.2)CALL PRTPAS(IREC,N,IRTS(1,N))
              GO TO 1000
     C
     C              TYPE #5 RECORD
     C
           150 DECODE(INREC,133)(IBCOND(K,N),K=1,IQPN)
          IF(IPRINT.GE.2)CALL PRTPAS(IREC,N,IBCOND(1,N))
              GO TO 1000
     C
     C              TYPE #6 RECORD
     C
           160 DECODE(INREC,133) (INRTS(K,N),K=1,IQPN)
          IF(IPRINT.GE.2)CALL PRTPAS(IREC,N,INRTS(1,N))
              GO TO 1000
     C
     C              TYPE #7 RECORD
     C
           170 DECODE(INREC,133) (IDRGN(K,N),K=1,IQPN)
          IF(IPRINT.GE.2)CALL PRTPAS(IREC,N,IDRGN(1,N))
              GO TO 1000
     C
     C              TYPE #8 RECORD
     C
```

```
      180 DECODE(INREC,133) (IDCOND(K,N),K=1,IQPH)
      IF(IPRINT.GE.2)CALL PRTPAS(IREC,N,IDCOND(1,N))
          GO TO 1000
C
C                TYPE #9 RECORD
C
      190 DECODE(INREC,133) (IDREP(K,N),K=1,IQPH)
      IF(IPRINT.GE.2)CALL PRTPAS(IREC,N,IDREP(1,N))
          GO TO 1000
C
C                TYPE #10 RECORD
C
      200 DECODE(INREC,133) (IOVCND(K,N),K=1,IQPH)
      IF(IPRINT.GE.2)CALL PRTPAS(IREC,N,IOVCND(1,N))
          GO TO 1000
C
C                TYPE #11 RECORD
C
      210 DECODE(INREC,133) (IPPROG(K,N),K=1,IQPH)
      IF(IPRINT.GE.2)CALL PRTPAS(IREC,N,IPPROG(1,N))
          GO TO 1000
C
C                READ IN NEW RECORD OFF OF FILE #7
C
 1000 CONTINUE
      READ(7,230,END=3000)ISEQ,IC1,IC2,IREC,INREC
      DECODE(INREC,233)ISNREC
  233 FORMAT(A100)
  230 FORMAT(I6,I1,I1,I2,A182)
C            TEST FOR CHANGES
      IF((IOLDSQ.NE.ISEQ).AND.(IREC.NE.1))
&       WRITE(6,243)"WARNING--",
&                "IOLDSQ=ISEQ AND IREC=1"
  243 FORMAT(V)
      IF((IOLDSQ.NE.ISEQ).AND.(IREC.EQ.1))GO TO 2000
C
C
C
C            PROCESS THIS RECORD
C
      GO TO 100
C
C            NEW RECORD IS AN 01
C
 2000 CONTINUE
      IF(IPRINT.LT.3) GO TO 2090
              WRITE(6,127)
&     UI,LTADN(N),LTPROD(N),IBRT(N),IDRCD,BPCODF,SMCODE,MPCODE
  127 FORMAT(//" UI  LTADN LTPROD   IBRT  IDRCD BPCODE SMCODE MPCODE",
&       /1X,A2,4X7,A7,A7,A7)
              WRITE(6,128)
&             IOST(N),IOINSL,NJRSL,IDFSL,OIMDDR(N)
```

```
      128 FORMAT(" IOST IOIMSL NJRSL IDFSL OIMBDB"/
     &            I6,I6,I7,I7,3X,F7.3 )
          WRITE(6,129)
     &            RTSF(N),PDMF(N),EOMF(N),RMSTRF(N),BPDDB(N),BMRTSF
      129 FORMAT(" RTSF PDMF EOMF RMSTRF DPDDB BMRTSF"/
     &            F7.2, F7.2,F7.2,  F7.2,  F6.0,  F7.0 )
          WRITE(6,139) PJRCND(N),PNJRC(N),OIMBRB(N),BCONF(N),
     &            EJCNF(N),ENCNF(N)
      139 FORMAT(' PJRCND PNJRC OIMBRB BCONF EJCNF ENCNF '/
     &            6F7.2 )
          WRITE(6,149)DOMCNF(N),RMSCNF(N),RMNRF(N),NAPPL,
     &            NUSERS,IENDAT
      149 FORMAT(' DOMCNF RMSCNF RMNRF NAPPL NUSERS IENDAT'/
     &         3F7.2,2I6,2X,I7 )
          WRITE(6,159)IESSN,RCOST(N),IRPHRS,JRSLD,IBPD(N),IRINTR(N),
     &            ISORM
      159 FORMAT(' IESSN RCOST IRPHRS JRSLD IBPD IRINTR ISORM'/
     &            ,2X,A4,F10.2,I4,4I7 )
          WRITE(6,169) ISFLOW,ISRTIN,IRLT,IPBD,IQPA(N),IAP
      169 FORMAT(' ISFLOW ISRTIN IRLT IPBD IQPA IAP'/
     &            6I7)
     2090 CONTINUE
C
C           LIMIT INPUT PARAMETERS TO "REASONABLE" VALUES
C
          IF(IOST(1).LE.0)IOST(1)=14
C
          IF(IQPA(N).LT.1)IQPA(N)=1
          IF(LTADM(N).LE.0)LTADM(N)=3
          IF(LTPROD(N).LE.0)LTPROD(N)=9
          IF(IBPD(N).LE.0)IBPD(N)=6
          IF(ISFLOW.LE.0)ISFLOW=30
          IF(IDRCD.LE.0)IDRCD=63
          IF(IDRCD.LE.ISFLOW)IDRCD=ISFLOW+2
C
C           SET TIME PARAMETERS FOR ITEM N
C
          IBRT(N)=IBPD(N)
          IDRT(N)=ISFLOW
          IBDTT(N)=IDRCD-ISFLOW
          IDORT(N)=ISFLOW
          IBDORD=1
          IOVSHP=IOST(1)-IBDORD
C
C           CONVERT PROCUREMENT LEADTIMES FROM MONTHS TO DAYS
C
          LTADM(N)=30*LTADM(N)
          LTPROD(N)=30*LTPROD(N)
C
C           SET ORDER AND SHIP TIME PARAMETERS
C
           DO 2220 K=1,NBASES
```

```
      JOST=IOST(1) + OSTDLT(K) + 0.5
      IDSHIP(K)=JOST-IBDORD
      IF(IDSHIP(N).LT.1)IDSHIP(N)=1
 2220 CONTINUE
C
C
      IC3=0
      IF((IWT(2).EQ.1).AND.(NGROUP.GE.NFGRP))
     &  WRITE(6,2223)JC1,JC2,JC3,JSEQ,FSN(N),UCCST(N)
 2223 FORMAT("SKU DATA=",3I6,A15,F10.2)
      IF((IOUT.EQ.1).AND.(NGROUP.GE.NFGRP))
     &  WRITE(8)JC1,JC2,JC3,JSEQ,FSN(N),UCOST(N)
     &          ,IOPA(N),IBDTT(N),IDRT(N),IBRT(N),IDORT(N),
     &        LTADM(N),LTPROD(N),IBDORD,IOVSHP,IDSHIP
      IOLDSQ=ISEQ
C
C          IF IWT(2)>= 2, PRINT OUTPUT RECORD HEADER
C
      IF( (IWT(2).GE.1).AND.(NGROUP.GE.NFGRP) )
     &  WRITE(6,2023)JC1,JC2,JC3,JSEQ,
     &          FSN(N),UCOST(N),IOPA(N),IBDTT(N),IDRT(N),IBRT(N),
     &          IDORT(N),LTADM(N),LTPROD(N),IBDORD,IOVSHP,IDSHIP
 2023 FORMAT(//" JC1 JC2 JC3 JSEQ          FSN            UCOST",
     &          "   IOPA  IBDTT   IDRT   IBRT  IDORT  LTADM LTPROD",
     &          " IBDORD IOVSHP"/
     &          3I4,I5,A16,F10.2,9I7//"   IDSHIP 1,2,...."/8I4)
      IF(IPRINT.GT.0) WRITE(6,179)
  179 FORMAT("0",50(1H,)//)
      IF(IC1.NE.3) GO TO 300
      N=N+1
      NITEM=N
      GO TO 100
C
C          IF IOUT=1, OUTPUT HEADER RECORD TO EXOGFILE
C           IF IWT(2)=1, ALSO PRINT HEADER RECORD
  300 CONTINUE
      WFSN="EVENTS FOLLOW"
      IF(NGROUP.LT.NFGRP) GO TO 320
      IF(IWT(2).EQ.1)WRITE(6,2223)0,0,0,IOLDSQ,WFSN,0.
      IF(IOUT.EQ.1)WRITE(8)0,0,0,IOLDSQ,WFSN,0.,IZERO
  320 CONTINUE
      RETURN
C
C
C          READ END OF FILE 7
 3000 CONTINUE
      IENDF=1
      RETURN
C
      ENTRY READF1
      REWIND 07
      READ(7,193,END=3000)ISEQ,IC1,IC2,IREC,INBEC
```

```
299          DECODE(INREC,233)ISNREC
300      193 FORMAT(I6,I1,I1,I2,A182)
301         IOLDSQ=ISEQ
302         RETURN
303         END
***W     7 MEMORY EXPANDED, USE $LIMITS OR CORE= OPTION FOR NEXT RUN
```

Subroutine: SCHIND

Function:

This routine schedules independent SRU Events.

Calling Parameters:

K	=	The SRU involved.
NUMB	=	The number of events of the current type to be generated.
KTYPE	=	The event type to be generated.
KQTR	=	The quarter in which the events are to be generated.
KLOC	=	Event location code. If KLOC equals 0, depot rep gen events are scheduled; otherwise, the events are scheduled at a randomly selected base.

Description:

Subroutine SCHIND first establishes a Stock Keeping Unit number for this rep gen. If KLOC equals 0, the SKU is set equal to the Stock Keeping Unit number of the depot location for SRU K. Otherwise, the SKU is set equal to the Stock Keeping Unit number for SRU K at a randomly selected base. The reparable generation counter NJOB is then incremented, and a Monte Carlo process is employed to establish at a randomly selected time within quarter KQTR. Sub-

routine ENTER is then called to place the reparable generation event (Event Type 14) and an associated requisition for a serviceable replacement (Event Type 1) on the Future Event List. The program logic then branches depending on the type (KTYPE) of event to be generated.

If a base condemnation or base repair completion is to be scheduled, subroutine ENTER is called to place appropirate Type 15 or Type 18 events on the Future Events List. On the other hand, if a depot reparable generation is involved, a Monte Carlo process is used to determine if the asset is to be depot condemned. If so, a depot condementation event (Event Type 15) is scheduled to occur at the depot location. Otherwise, a NRTS event (Event Type 19) is scheduled to simulate shipment of the asset to the depot. In addition, when a NRTS event is scheduled, a Monte Carlo process is used to determine if the NRTS asset is to be condemned at the depot, or if the asset will be reparable at that location. If the asset is to be condemned at the depot, an appropriate condemnation event (Event Type 15) is scheduled to occur at the depot after an appropriate transportation time. Otherwise, a repair completion event (Event Type 18) is scheduled to occur after a time delay which allows for the transportation of the asset back to the depot, and the repair of the asset at that location.

In all of the above cases, appropriate counters of SRU events are reset after each of the Monte Carlo calculations to ensure that the correct probability values are used in subsequent Monte Carlo computations.

T 01 10-20-79 10.349

```
                SUBROUTINE SCHIND(K,NUMB,KTYPE,KQTR,KLOC)
      C
      C                 THIS ROUTINE SCHEDULES NUMB INDEPENDENT KTYPE EVENTS
      C                 FOR SRU K TO OCCUR IN QUARTER KQTR.
      C                 IF KLOC=0,DEPOT REP GEN EVENTS ARE SCHEDULED
      C            OTHERWISE, THE
      C                 EVENTS ARE SCHEDULED AT A RANDOMLY SELECTED BASE.
            COMMON/NOYCNT/NOYCNT(40)
            COMMON/NOYGNT/NOYGNT(40)
            COMMON/NODRPT/NODRPT(40)
            COMMON/NDRGN/NDRGN(40)
            COMMON/NDRGNT/NDRGNT(40)
            COMMON/NDCONT/NDCONT(40)
            COMMON/NDCOND/NDCOND(40)
      C
             COMMON/IDRT/IDRT(40)
             COMMON/IBDTT/IBDTT(40)
             COMMON/IBRT/IBRT(40)
      C
            COMMON/IWT/IWT(20)
      C
            COMMON/NSRU/NSRU
                  COMMON/NJOB/NJOB
                  COMMON/ITIME/ITIME
                  COMMON/NBASES/NBASES
                  COMMON/ITDAY/ITDAY
                  COMMON/ITQTR/ITQTR
      ***********************************
      C
            IDBUG=INT(6)
      C
            MMMM=NUMB
             DO 90  NNN=1,MMMM
      C
      C                 ESTABLISH THE STOCKKEEPING NUMBER FOR THIS SER GEN
      C
            IFSN=K+1
            NSKU=(NBASES+2)*(K+1)
            CALL RANDES(KBASE)
                  IF(KLOC.NE.0)NSKU=K*(NBASES+2) +1 + KBASE
            IF(IDBUG.EQ.1) WRITE(6,13)
      &           "****SCHIND--K,NUMB,KTYPE,KQTR,KLOC,NSKU"
         13 FORMAT(V)
            IF(IDBUG.EQ.1)WRITE(6,23) K,NUMB,KTYPE,KQTR,KLOC,NSKU
         23 FORMAT(10I7)
      C
      C                 INCREMENT NJOB
               NJOB=NJOB+1
      C
      C                 MONTE CARLO TO ESTABLISH EVENT TIME
      C
                ITIME=(KQTR-1)*ITQTR+RAND0(.2)*ITQTR
```

SCHIND

```
C
C                    SCHEDULE REP GEN EVENT
C
                JTIME=ITIME
                CALL ENTER(JTIME,10,NSKU,9,NJOB)
C
C                    SCHEDULE REQUISITION FOR REPLACEMENT UNITS
C
                JTIME=ITIME+10
                CALL ENTER(JTIME,1,NSKU,1E1)
C
C                    BRANCH BY EVENT TYPE
C                    DEPOT GENERATION=14
C                    CONDEMNATION =15,RTO=18,NRTS=19, OTHERS ARE ERROR
C
                IF(KTYPE.EQ.14) GO TO 300
                IF(KTYPE.EQ.15) GO TO 100
                IF(KTYPE.EQ.18) GO TO 200
                IF(KTYPE.EQ.19) GO TO 400
                PRINT,"***SCHIND--ILLEGAL EVENT TYPE=",KTYPE,K,
     &          NUMB,KOTR,KLOC
                STOP
C
C                    CONDEMNATION EVENT
C
           10   CONTINUE
                JTIME=ITIME +30
                CALL ENTER(JTIME, 15,NSKU,1,NJOB)
                GO TO 500
C
C                    RTS EVENT
C
           20   CONTINUE
                JTIME=ITIME +IBRT(IFSN)*IRDAY
                CALL ENTER(JTIME,18,NSKU,9,NJOB)
                GO TO 500
C
C                    MONTE CARLO TO DETERMINE IF SRU IS DEPOT CONDEMNED.
C
         300 CONTINUE
             DCONR=FLOAT(NDCOND(K))/FLOAT(NDRGN(K))
             R=RANDU(1,2)
             IF(IDEVE.EQ.1)WRITE(6,13) "<<<<<<<SRU DEPOT REP GEN",
     &        "   NDCOND=",NDCOND(K),"   NDRGN=",NDRGN(K),
     &        "   DCONR=",DCONR,"   R=",R
C
C                    UPDATE DEPOT REP GEN COUNTS
C
             NDRGN(K)=NDRGN(K)+1
             NDRGNT(K)=NDRGNT(K)-1
C
             IF(R.GT.DCONR) GO TO 400
```

```
      C
      C          SKU IS DEPOT CONDEMNED
      C
            NDCOND(K)=NDCOND(K)-1
            NDCONT(K)=NDCONT(K)-1
            JTIME=ITIME+10
            CALL ENTER(JTIME,15,NSKU,1,NJOB)
            GO TO 500
      C
      C                 NRTS EVENT
      C
            40  CONTINUE
            NOVCNT(K)=NOVCNT(K)-1
                 JTIME=ITIME +30
                 CALL ENTER(JTIME,19,NSKU,5,NJOB)
      C
      C          COMPUTE DEPOT STOCK KEEPING NUMBER, AND MONTE CARLO TO
      C               DETERMINE IF THIS UNIT IS REPARABLE.
      C
      C
            NSKDEP=(NBASNS+2)*K + 1
            DCONR=FLOAT(NOVCNT(K )/FLOAT(NOVCNT(K )+NODRPT(K ))
            R=RANDU(0,0)
      C
            IF(IDBUG.EQ.1)WRITE(6,13)"<<<<<DEPOT LEVEL MAIN.=",
      &        "NOVCNT=",NOVCNT(K),"    NODRPT=",NODRPT(K),
      &        "    DCONR=",DCONR,"    R=",R
      C
      C          IF R IS LESS THAN OR EQUAL TO DCONR,UNITS IS CONDEMNED
      C
            IF(R.LE.DCONR)GO TO 490
      C
      C          NRTS IS REPARABLE,SCHEDULE REPAIR COMPLETION
      C
            NODRPT(K)=NODRPT(K)-1
            JTIME=ITIME+(IDDTT(IFSN)+IDRT(IFSN))*ITDAY
            CALL ENTER(JTIME,18,NSKDEP,1,NJOB)
            GO TO 500
      C
      C          SKU IS CONDEMNED AT THE DEPOT
      C
        490 CONTINUE
            NOVCNT(K)=NOVCNT(K)-1
            JTIME=ITIME+(IDDTT(IFSN)*ITDAY)
            CALL ENTER(JTIME,15,NSKDEP,1,NJOB)
            NOVCNT(K )=NOVCNT(K )-1
      C
        500 CONTINUE
        900 CONTINUE
      C
            IF(IDBUG.EQ.1)WRITE(6,13)"**WRIT SCHEND"
      C
```

```
T 01   10-20-79   10.949

57            RETURN
58            END
```

Subroutine: SRUEVT

Function:

This routine schedules all SRU events associated with the current LRU reparable generation.

Calling Parameters:

LRUREP	=	LRU Repairability Flag. LRUREP = 0 indicates the LRU is to be condemned, while RUREP = 1 indicates the LRU is reparable.
NSKLRU	=	The Stock Keeping Unit number for the LRU reparable generation.

Description:

For a detailed discussion of the functions of this routine, see the Event Generator Chapter of Volume I.

```
*$RUN*:RIME/OBJ/SRUEVT.O(BCD,NOGO)
*SRUEVT.S  GENERATE SRU EVENTS
      SUBROUTINE SRUEVT(LRUREP,NSKLRU)
C          SCHEDULE SRU EVENTS ASSOCIATED WITH THE CURRENT LRU
C            REP GEN. WHERE LRUREP=0 INDICATES LRU IS CONDEMNED.
C                              LRUREP=1 INDICATES LRU IS REPARABLE.
C
C                            NSKLRU= STOCK KEEPING UNIT FOR LRU REP GEN.
C                                    WHERE 1 DENOTES THE DEPOT.
C
      COMMON/IWT/IWT(20)
      COMMON/NJOB/NJOB
      COMMON/NSRU/NSRU
      COMMON/IOPA/IOPA(40)
      COMMON/NOINL/NOINL(40)
      COMMON/NBRTS/NBRTS(40)
      COMMON/NDCONT/NDCONT(40)
      COMMON/NDRGNT/NDRGNT(40)
      COMMON/NNRTS/NNRTS(40)
      COMMON/NBCOND/NBCOND(40)
      COMMON/NOVCNT/NOVCNT(40)
      COMMON/NOVGNT/NOVGNT(40)
      COMMON/NODRPT/NODRPT(40)
      COMMON/NOSRUF/NOSRUF(40)
C
       COMMON/IDRT/IDRT(40)
       COMMON/IBDTT/IBDTT(40)
       COMMON/IBRT/IBRT(40)
C
      COMMON/ITOTR/ITOTR
       COMMON/ITDAY/ITDAY
       COMMON/NBASES/NBASES
       COMMON/ITINV/ITINV
      COMMON/ITIME/ITIME
C
      IF(IWT(6).EQ.1)WRITE(6,13)*****SRUEVT--LRUREP=*,
     &      LRUREP,*   NSKLRU=*,NSKLRU
   13 FORMAT(V)
C
C
C
C                    BEGIN SRU LOOP
C
       DO 400 K=1,NSRU
C
C            SET INDEX FOR SRU TIME REFERENCES
C
      IFSN=K+1
C
C             IF THERE ARE NO FAILURES OF THIS SRU, GO TO  400
C
      IF(NOINL(K).LE.0) GO TO 400
```

```
C
C          COMPUTE THE  STOCK KEEPING NUMBER FOR SRU(K) AT BASE NB-
C
      NSKSRU=(NBASES+2)*K+NSKLRU
C
C          RECORD GENERATION OF NOINL(K) UNITS OF SRU K
C
      JTIME=ITIME + 15
      NOINLL=NOINL(K)
      CALL ENTER(JTIME,14,NSKSRU,NOINLL,NJOB)
C
C
C          IF LRU IS TO BE CONDEMNED(LRUREP=0),GO TO 160.
C
      IF(LRUREP.LE.0) GO TO 160
C
C          SCHEDULE A REQUISITION TO REPLACE THE NOINL(K) UNITS.
C
      JTIME=ITIME +20
      LPRI=(100*NJOB+2)
      CALL ENTER(JTIME,1,NSKSRU,NOINLL,LPRI)
      GO TO 190
C
C          SINCE LRU IS CONDEMNED, SCHEDULE SERVICEABLE RETURNS
C            (EVT CODE 4), IF ANY.  THEN GO TO 190.
C
  160 CONTINUE
      NS=IOPA(K)-NOINL(K)
      JTIME=ITIME+20
      IF(NS.GT.0) CALL ENTER(JTIME,4,NSKSRU,NS,NJOB)
C
  190 CONTINUE
C
C           IF THERE ARE NO SRU K EVENTS, GO TO 310
      IF(NOINL(K).LE.0) GO TO 310
C
C
C
C          FOR EACH RELATED SRU FAILURE, LOOP FOR KK=1,2,....
C          NOINL(K).
C
      DO 300KK=1,NOINL(K)
C
C            IF THIS IS A BASE LRU REP GEN. GO TO 240
C
      NSKOVH=NBASES+2
      IF( (NSKLRU.GT.1) .AND. (NSKLRU.LT.NSKOVH)) GO TO 240
C
C          REPARABLE LRU IS AT DEPOT
C               DECREMENT COUNT OF OVERHAUL GENERATIONS
      NOVGNT(K)=NOVGNT(K)-1
C
```

```
      C
      C          MONTE CARLO TO DETERMINE IF FAILURE KK IS DEPOT
      C              CONDEMNED.
      C
            DCONR=FLOAT(NDCONT(K))/FLOAT(NDRGNT(K))
            R=RANDU(0,8)
      C
            IF(IWT(6).EQ.1)WRITE(6,13)"    UNIT=",KK,"   NDCONT=",NDCONT(K),
     &          "     NDRGNT=",NDRGNT(K),"   DCONR=",DCONR,"   R=",R
      C
            IF(R.LE.DCONR)GO TO 230
      C
      C          SCHEDULE SRU REPAIR COMPLETION
      C
            JTIME=ITIME+IBRT(IPSN)*ITDAY
            CALL ENTER(JTIME,18,NSKSRU,1,NJOB)
            NDRGNT(K)=NDRGNT(K)-1
            GO TO 300
      C
      C          SCHEDULE SRU CONDEMNATION
      C
        230 CONTINUE
            JTIME=ITIME + 30
            CALL ENTER(JTIME,15,NSKSRU,1,NJOB)
            NDCONT(K)=NDCONT(K)-1
            NDRGNT(K)=NDRGNT(K)-1
            GO TO 300
      C
      C          MONTE CARLO TO DETERMINE IF FAILURE KK IS RTS, NRTS O
      C          ORCONDEMMED.
      CC
        240 CONTINUE
            PRTS=FLOAT(NBRTS(K))/FLOAT(NBRTS(K)+NNRTS(K)+NBCOND(K))
            CRTS=FLOAT(NBCOND(K))/FLOAT(NBRTS(K)+NNRTS(K)+NBCOND(K))+PRTS
            R=RANDU(0,7)
            IF(IWT(6).EQ.1)WRITE(6,313)KK,PRTS,CRTS,R
        313 FORMAT("     UNIT=",I5,"   PRTS=",F5.3,"   CRTS=",F5.3,"  R=",F5.3)
      C
      C          IF R IS <= PRTS, FAILURE IS AN RTS.
      C
            IF(R.LE.PRTS) GO TO 250
      C
      C     IF R IS LESS THAN OR EQUAL TO CRTS, FAILURE IS CONDEMNATION.
      C
            IF  (R.LE.CRTS)GO TO 270
      C
      C          IF R IS GREATER THAN CRTS, FAILURE IS A NRTS.
      C          SCHEDULE NRTS EVENT, DECREMENT NRTS COUNTER.
      C
            JTIME=ITIME+30
            CALL ENTER(JTIME,19,NSKSRU,1,NJOB)
            NNRTS(K )=NNRTS(K )-1
```

```
      GO TO 280
C
C           SCHEDULE RTS EVENT, AND DECREMENT RTS COUNTER.
C
  250 JTIME=ITIME+IBRT(IFSN)*ITDAY
      CALL ENTER(JTIME,18,NSKSRU,1,NJOB)
      NRRTS(K )=NRRTS(K )-1
      GO TO 300
C
C           SCHEDULE COND EVENT, AND DECREMENT COND COUNTER.
C
  270 CONTINUE
      JTIME=ITIME + 30
      CALL ENTER(JTIME,15,NSKSRU,1,NJOB)
      NBCOND(K )=NBCOND(K )-1
      GO TO 300
C
C           IF FAILURE IS NRTS, MONTE CARLO TO DETERMINE IF THE SRU
C           IS TO BE REPAIRED OR CONDEMNED AT THE DEPOT, THEN SCHEDULE
C           THE CORRESPONDING REPAIR COMPLETION OR CONDEMNATION
C           EVENT.
C
  280 CONTINUE
C           DECREMENT COUNT OF OVERHAUL GENERATIONS
C
      NOVGNT(K)=NOVGNT(K)-1
      NSKDEP=(NBASES+2)*K + 1
      DCONR=FLOAT(NOVCNT(K ))/FLOAT(NOVCNT(K )+NCDRPT(K ))
      R=RANLU(0.8)
C
C           IF R IS LESS THAN OR EQUAL TO DCONR,NRTS IS CONDEMNED
C
      IF(R.LE.DCONR)GO TO 290
C
C           NRTS IS REPARABLE,SCHEDULE REPAIR COMPLETION
C
      IFSN=K+1
      JTIME=ITIME+(IBDTT(IFSN)+IDRT(IFSN))*ITDAY
      CALL ENTER(JTIME,18,NSKDEP,1,NJOB)
      NOBRPT(K )=NOBRPT(K )+1
      GO TO 300
C
C           SRU IS CONDEMNED AT THE DEPOT
C
  290 CONTINUE
      IFSN=K+1
      JTIME=ITIME + (IBDTT(IFSN)*ITDAY)
      CALL ENTER(JTIME,15,NSKDEP,1,NJOB)
      NOVCNT(K )=NOVCNT(K )-1
C
C           END OF KK FAILURE LOOP
C
```

```
  300 CONTINUE                                                          60
  310 CONTINUE
C
C           END OF SRU(K) LOOP
C
  400 CONTINUE
  500 CONTINUE
      IF(IWT(6).EQ.1) WRITE(6,13)"------EXIT SRUEVT"
      RETURN
    END
```

Subroutine: SRUIND

Function:

This routine scans the SRU event counter arrays, and schedules independent SRU events as needed.

Description:

This routine scans the events counter arrays for each SRU. The routine is called after all LRU events have been generated. If any of the SRU arrays are non-zero, the remaining SRU events are assumed to occur independently of LRU reparable generations. Consequently, subroutine SRUIND places appropriate events on the Future Events List to simulate these independent SRU events.

If the counter of SRU base RTS events (NBRTS(K)) for SRU K is greater than zero, subroutine SCHIND is called to place NBRTS(K) type 18 events on the Future Events List for SRU K. Similarly, if the SRU event counters NBCOND(K), NNRTS(K), or NDRGN(K) is greater than zero for any SRU K, subroutine SCHIND is called to schedule the associated type 15, 19, and 14 events, respectively.

```
      SUBROUTINE SRUPRT
         COMMON/NSRU/NSRU
         COMMON/IQPA/IQPA(40)
         COMMON/NOINL/NOINL(40)
         COMMON/NBRTS/NBRTS(40)
         COMMON/NNRTS/NNRTS(40)
         COMMON/NBCOND/NBCOND(40)
         COMMON/NOVCNT/NOVCNT(40)
         COMMON/NODRPT/NODRPT(40)
         COMMON/NOSRUF/NOSRUF(40)
         COMMON/NDRGN/NDRGN(40)
         COMMON/NDRGNT/NDRGNT(40)
         COMMON/NDCOND/NDCOND(40)
          COMMON/NDCONT/NDCONT(40)
         COMMON/NOVGNT/NOVGNT(40)
         COMMON/NDEXPO/NDEXPO(40)
         WRITE(6,413)
     413 FORMAT(  'NJ',T10,'  IQPA',T20,'  NOINL',T30,
     &              '  NBRTS',T40,'  NNRTS',T50,'  NBCOND',
     &            T60,'  NOVCNT',T70,'  NODRPT',T80,'  NOSRUF')
         DO 430 NJ=1,NSRU
         WRITE(6,423) NJ,IQPA(NJ),NOINL(NJ),NBRTS(NJ),
     &                  NNRTS(NJ),NBCOND(NJ),NOVCNT(NJ),
     &                NODRPT(NJ),NOSRUF(NJ)
     423 FORMAT ( I3,T6,8I10)
     430 CONTINUE
     WRITE(6,433)
     433 FORMAT(  'NJ',T10,'  NDRGNT',T20,'  NDCONT',T30,
     &                '  NOVGNT',T40,'  NDEXPO',T50,'   NDRGN',
     &                T6 ,'  NDCOND')
         DO 437 NJ=1,NSRU
         WRITE(6,436) NJ,NDRGNT(NJ),NDCONT(NJ),NOVGNT(NJ),
     &                NDEXPO(NJ),NDRGN(NJ),NDCOND(NJ)
     436 FORMAT ( I3,T6,10I10)
     437 CONTINUE
     RETURN
     END
```

2T 01 10-20-79 10.349 S

```
*$RUN=LRTME/OBJ/SRUIND.O(BCD,NOGO)
*SRUIND.S
                    SUBROUTINE SRUIND
C                    SCAN SRU EVENT COUNTER ARRAYS, AND SCHEDULE INDEPENDENT
C                    SRU EVENTS AS NEEDED.
C
      COMMON/NSRU/NSRU
      COMMON/NBRTS/NBRTS(40)
      COMMON/NBCOND/NBCOND(40)
      COMMON/NNRTS/NNRTS(40)
      COMMON/NDRGN/NDRGN(40)
      COMMON/ITINV/ITINV
C
C
            DO 900 KK=1,NSRU
            K=KK
C
C                    SCHEDULE RTS EVENTS (CODE 18)?
C
             KQTR=ITINV
            IF(NBRTS(K).GT.0)CALL SCHIND(K,NBRTS(K),18,KQTR,1)
C
C                    SCHEDULE CONDEMNATION EVENT (CODE15)?
C
            IF(NBCOND(K).GT.0)CALL SCHIND(K,NBCOND(K),15,KQTR,1)
C
C                    SCHEDULE NRTS EVENT A(CODE19)?
C
            IF(NNRTS(K).GT.0)CALL SCHIND(K,NNRTS(K),19,KQTR,1)
C
C                    SCHEDULE DEPOT REP GENS, IF ANY
C
            IF(NDRGN(K).GT.0) CALL SCHIND(K,NDRGN(K),14,KQTR,0)
      900 CONTINUE
            RETURN
            END
```

SRUIND

T 01 11-07-79 14.524

```
      SUBROUTINE SRUPRT
         COMMON/NSRU/NSRU
         COMMON/IQPA/IQPA(40)
         COMMON/NOINL/NOINL(40)
         COMMON/NBRTS/NBRTS(40)
         COMMON/NNRTS/NNRTS(40)
         COMMON/NBCOND/NBCOND(80)
         COMMON/NOVCNT/NOVCNT(80)
         COMMON/NODRPT/NODRPT(80)
         COMMON/NOSRUF/NOSRUF(80)
         COMMON/NDRGN/NDRGN(40)
         COMMON/NDRGNT/NDRGNT(80)
         COMMON/NDCOND/NDCOND(80)
          COMMON/NDCONT/NDCONT(40)
         COMMON/NOVGNT/NOVGNT(80)
         COMMON/NDEXPO/NDEXPO(40)
         WRITE(6,413)
     413 FORMAT(  'NJ',T10,'  IQPA',T20,'  NOINL',T30,
     &            '  NBRTS',T40,'  NNRTS',T50,'  NBCOND',
     &            T60,'  NOVCNT',T70,'  NODRPT',T80,'  NOSRUF')
         DO 430 NJ=1,NSRU
         WRITE(6,423) NJ,IQPA(NJ),NOINL(NJ),NBRTS(NJ),
     &                NNRTS(NJ),NBCOND(NJ),NOVCNT(NJ),
     &               NODRPT(NJ),NOSRUF(NJ)
     423 FORMAT ( I3,T6,8I10)
     430 CONTINUE
     WRITE(6,433)
     433 FORMAT(  'NJ',T10,'  NDRGNT',T20,'  NDCONT',T30,
     &            '  NOVGNT',T40,'  NDEXPO',T50,'  NDRGN',
     &            T60,'  NDCOND')
         DO 437 NJ=1,NSRU
         WRITE(6,436) NJ,NDRGNT(NJ),NDCONT(NJ),NOVGNT(NJ),
     &               NDEXPO(NJ),NDRGN(NJ),NDCOND(NJ)
     436 FORMAT ( I3,T6,10I10)
     437 CONTINUE
     RETURN
     END
```

REFERENCES

1. Demmy, W. Steven, RIME: The Recoverable Item Management Evaluator, Volume I: Model Description, TR-80-01, Decision Systems, 3575 Charlene Drive, Dayton, Ohio 45432, May 1980, 153 pp.

2. Demmy, W. Steven, RIME: The Recoverable Item Management Evaluation Volume II, Program Listings and Narratives, TR-80-02, Decision Systems, 3575 Charlene Drive, Dayton, Ohio 45432, May 1980, 298 pp.

3. Demmy, W. Steven, An Empirical Evaluation of Proposed Stockage Policies for Recoverable Item Management, TR-80-03, Decision Systems, 3575 Charlene Drive, Dayton, Ohio 45432, May 1980, 173 pp.

4. Demmy, W. Steven and Victor J. Presutti, Jr., Multi-Echelon Inventory Theory in the Air Force Logistics Command, Working Paper 76-3011-27, Department of Management, Wright State University, Dayton, Ohio 45435.

5. Muckstadt, John A., "A Model for a Multi-Item, Multi-Echelon, Multi-Indenture Inventory System, Management Science, v20, n4, December, 1973, pp. 472-481.