

***MODEL DOCUMENTATION
LOAD AND DEMAND SIDE
MANAGEMENT SUBMODULE***

*Prepared for Energy Information Administration
1000 Independence Avenue, S.W.
Washington, DC 20585.*

*Submitted by
Decision Analysis Corporation of Virginia
and
ICF Resources Incorporated.*

Contract No: DE-AC01-92E121946

VOLUME II
MODEL CODE

TABLE OF CONTENTS

	Page
1. INTRODUCTION	1
2. SOURCE CODE OF THE LDSM SUBMODULE	2
3. INCLUDE FILES USED WITH THE LDSM SOURCE CODE	71
4. CROSS REFERENCE TABLE OF SUBMODULE VARIABLES	84

1. INTRODUCTION

Volume II of the documentation contains the actual source code of the LDSM submodule, and the cross reference table of its variables. The code is divided into two parts. The first part contains the main part of the source code. The second part lists the INCLUDE files referenced inside the main part of the code.

LDSM Source-Code

```
line 1: SUBROUTINE ELLDSM(LDSMmode)
line 2: C***** Description of the Program/Subprogram *****
line 3: C This is the main subroutine of LDSM module
line 4: C-----
line 5: C Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 6: C*****
line 7: IMPLICIT NONE
line 8: C***** Typing, Declaring and Initializing Constant Parameters ****
line 9: INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 10: INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 11: C***** Typing and Declaring Variables *****
line 12: INTEGER*4 LDSMmode ! -1 - prepare data for ECP every iteration
line 13: ! 1 - prepare data for ECP only on first iteration
line 14: ! 2 - process ECP solution
line 15: LOGICAL WHOOPS ! ERROR FLAG
line 16: INTEGER TMRPNB
line 17: C***** COMMON and EQUIVALENCE *****
line 18: INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 19: INCLUDE 'DSMFMGRD' !<< FILE_MGR variables declarations
line 20: INCLUDE 'NCNTRL' !<< global NEMS variables
line 21: INCLUDE 'DSMNERCR' !<< NERC region data
line 22: INCLUDE 'DSMNEMSC' !<< results to be passed to the rest of NEMS
line 23: INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 24: INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 25: C***** Initializing Variables *****
line 26: WHOOPS=.FALSE.
line 27: C***** Body of the Program/Subprogram *****
line 28: C On the very first iteration of NEMS run:
line 29: TotDemNERC=0.0
line 30: IF(CURIYR.EQ.FIRSYR .AND. CURITR.EQ.1 .AND. LDSMmode.EQ.1) THEN
line 31: NEW=.TRUE.
line 32: fname='LDSMRPT'
line 33: IMSG=FILE_MGR('O',fname,NEW) !Open LDSM REPORT FILE
line 34: NEW=.FALSE.
line 35: fname='LDSMDAF'
line 36: IODB=FILE_MGR('O',fname,NEW) !Open DAF-LSR-DB
line 37: NEW=.FALSE.
line 38: fname='LDSMCRS'
line 39: IOCR=FILE_MGR('O',fname,NEW) !Open COMMERCIAL RESTART FILE
line 40: fname='LDSMRRS'
line 41: IORR=FILE_MGR('O',fname,NEW) !Open RESIDENTIAL RESTART FILE
line 42: NEW=.FALSE.
line 43: WRITE(MSG,*)'LDSM REPORT FILE (units: GW and GWh) .'
line 44: CALL DSMRST(WHOOPS) ! Read structure file and DSM option database
line 45: IF(WHOOPS) THEN
line 46: WRITE(6,*)'<) Message from LDSM module 1:'
line 47: WRITE(6,*)'<)) Processing interrupted because of an ERROR'
line 48: WRITE(6,*)'<)) Data passed by LDSM may be CORRUPTED'
line 49: WRITE(6,*)'<)) Control returned to UTIL'
line 50: RETURN
line 51: ENDIF
line 52: CALL DSMRESD ! SPECIFY RESIDENTIAL TECHNOLOGY DATA
line 53: ENDIF
line 54: WRITE(MSG,*)'<)) ITERATION NUMBER, YEAR NUMBER:',CURITR,CURIYR
line 55: WRITE(MSG,*)'<)) LDSMmode:',LDSMmode
line 56: IF (LDSMmode.LT.2) THEN ! LDSM is run to prepare data for EFP,ECP,EFD
line 57: CALL DSMCADJ! Calculate adjustment factors for COMMERCIAL LOAD
line 58: IF(CURIYR.GE.FIRSYR+2) THEN
line 59: CALL DSMEFP2! Calculate current year DSM costs for EFP
```

```

line 60:   ENDIF
line 61:   DO RNB=1,nNERCreg ! Do for each NERC region to be processed
line 62:       ! Because RNB is a parameter of a global type, used by
line 63:       ! many level of routines within the loop, its value
line 64:       ! is passed through a common block, rather than as
line 65:       ! explicit subroutine parameter
line 66:   K1=CURIYR
line 67:   WRITE(IMG,*)'~~~~~ NERC REGION: ',NERCnam(RNB),' ~~~~~'
line 68:   WRITE(*,*)'~~~~~ NERC REGION: ',NERCnam(RNB),' ~~~~~'
line 69: C Upload from direct access file desired year solution of ECP
line 70:   IF(CURIYR.GT.FIRSyr .AND. CURITR.EQ.1) THEN
line 71:     TmprNB=RNB
line 72:     CALL GETBOUT(CURIYR-1,TmprNB)
line 73:   ENDIF
line 74:   CALL DSMFOR(WHOOPS) ! Prepare load forecast for current year
line 75:   IF(WHOOPS) THEN
line 76:     WRITE(6,*)'<') Message from LDSM module 2:'
line 77:     WRITE(6,*)'<)) Processing interrupted because of an ERROR'
line 78:     WRITE(6,*)'<)) Data passed by LDSM may be CORRUPTED'
line 79:     WRITE(6,*)'<)) Control returned to UTIL'
line 80:     RETURN
line 81:   ENDIF
line 82:   CALL DSMHLM ! Run procedures which develop system load
line 83:   CALL DSMEFP ! Prepare data required by EFP
line 84:   CALL DSMTOR ! Prepare sectorial variables for NEMS REPORT WRITER
line 85:   CALL DSMEFD ! Develop LDC's for EFD
line 86:   IF(CURITR.EQ.1 .OR. LDSMmode .EQ. -1) THEN
line 87:     CALL DSMECP1(WHOOPS) !Develop LDC's and DSM Program data for ECP
line 88:     IF(WHOOPS) THEN
line 89:       WRITE(6,*)'<') Message from LDSM module 3:'
line 90:       WRITE(6,*)'<)) Processing interrupted because of ERROR'
line 91:       WRITE(6,*)'<)) Data passed by LDSM may be CORRUPTED'
line 92:       WRITE(6,*)'<)) Control returned to UTIL'
line 93:       RETURN
line 94:     ENDIF
line 95:   ENDIF
line 96:   ENDDO
line 97: ELSE
line 98:   CALL DSMECP2 ! Prepare DSM impact data for demand modules
line 99: ENDIF
line 100: C***** Termination of the Program/Subprogram *****
line 101:   WRITE(IMG,*)'<)) Execution successfully completed'
line 102:   RETURN
line 103: END
line 104: SUBROUTINE DSMACM(WHOOPS)
line 105: C***** Description of the Program/Subprogram *****
line 106: C This subroutine creates load forecasts for each end-use and NERC region
line 107: C using data imported from the COMMERCIAL sector demand forecasting module.
line 108: C-----
line 109: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 110: C*****
line 111:   IMPLICIT NONE
line 112: C***** Typing, Declaring and Initializing Constant Parameters ****
line 113:   INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 114:   INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 115:   INCLUDE 'NCNTRL' !<< global NEMS variables
line 116: C***** Typing and Declaring Variables *****
line 117:   REAL*4 load ! total load
line 118:   INTEGER*2 l,i,j,k ! universal counters
line 119:   LOGICAL WHOOPS ! error flag
line 120:   REAL*4 CENSUSvalues(MAXCRG) ! temporary vector with CENSUS division values
line 121:   REAL*4 FORESIGHTadj(MAXCRG) ! FORESIGHT adjustment factor

```

```

line 122:  EXTERNAL DSMNVAL
line 123: C***** COMMON and EQUIVALENCE *****
line 124:  INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 125:  INCLUDE 'COMPARM' !<< defines PARAMETERS: CMnumBldg,CMnumServ
line 126:  INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 127:  INCLUDE 'DSMNERCR' !<< NERC region data
line 128:  INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 129:  INCLUDE 'QBLK' !<< supplies some constants needed in MXQBLK
line 130:  INCLUDE 'MXQBLK' !<< foresight data
line 131: C***** Initializing Variables *****
line 132: C***** Body of the Program/Subprogram *****
line 133:  I=EUINDEX(SEC(COM),1)
line 134:  DO k=1,nCENSUSreg
line 135:    FORESIGHTadj(k)=XQELCM(k,K1)/XQELCM(k,CURIYR)
line 136:  ENDDO
line 137:  DO i=1,CMnumBldg
line 138:    DO j=1,CMnumServ
line 139: C Aggregate CENSUS regions load forecast into NERC region forecasts by end-use
line 140:    DO k=1,nCENSUSreg
line 141:      CENSUSvalues(k)=EndUseConsump(ELINDEX,j,i,k,CURIYR)
line 142:    & *ComDemAdjFac(k)*FORESIGHTadj(k)
line 143:    ENDDO
line 144:    LFinum=LFinum+1
line 145:    LoadForec(LFinum,1)=DSMNVAL(CENSUSvalues,RNB,SEC(COM))
line 146:    & *UNCONFA
line 147:    I=I+1
line 148:  ENDDO
line 149: ENDDO
line 150: I=I-1
line 151: IF(I.NE.EUINDEX(SEC(COM),2)) GOTO 999
line 152: C***** Termination of the Program/Subprogram *****
line 153:  RETURN
line 154: 999 WRITE(IMG,*)'<)) Message from subroutine DSMACM'
line 155:  WRITE(IMG,*)'<)) Wrong number of end-uses specified on ',
line 156:  &'LDSMSTRU file for COMMERCIAL sector, region: ',NERCnam(RNB)
line 157:  WHOOPS=.TRUE.
line 158:  RETURN
line 159:  END
line 160:  SUBROUTINE DSMAIN(WHOOPS)
line 161: C***** Description of the Program/Subprogram *****
line 162: C This subroutine creates load forecasts for each end-use and NERC region
line 163: C using data imported from the INDUSTRIAL sector demand forecasting module.
line 164: C-----
line 165: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 166: C*****
line 167:  IMPLICIT NONE
line 168: C***** Typing, Declaring and Initializing Constant Parameters ***
line 169:  INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 170:  INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 171:  INCLUDE 'NCNTRL' !<< global NEMS variables
line 172: C***** Typing and Declaring Variables *****
line 173:  REAL*4 load ! total load
line 174:  INTEGER*2 i,k,l ! universal counters
line 175:  LOGICAL WHOOPS ! error flag
line 176:  REAL*4 CENSUSvalues(MAXCRG) ! temporary vector with CENSUS division values
line 177:  REAL*4 FORESIGHTadj(MAXCRG) ! FORESIGHT adjustment factor
line 178:  EXTERNAL DSMNVAL
line 179: C***** COMMON and EQUIVALENCE *****
line 180:  INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 181:  INCLUDE 'QBLK' !<< access to QELIN matrix
line 182:  INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 183:  INCLUDE 'DSMNERCR' !<< NERC region data

```

```

line 184:  INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 185:  INCLUDE 'MXQBLK' !<< foresight data
line 186: C***** Initializing Variables *****
line 187: C***** Body of the Program/Subprogram *****
line 188:  I=EUINDEX(SEC(IND),1)
line 189:  DO k=1,nCENSUSreg
line 190:    FORESIGHTadj(k)=XQELIN(k,K1)/XQELIN(k,CURIYR)
line 191:  ENDDO
line 192: C Aggregate CENSUS regions load forecast into NERC region forecasts by end-use
line 193:  DO i=1,NEUSES(SEC(IND))
line 194:    DO k=1,nCENSUSreg
line 195:      CENSUSvalues(k)=QELIN(k,CURIYR)*FORESIGHTadj(k)
line 196:    ENDDO
line 197:    LFinum=LFinum+1
line 198:    LoadForec(LFinum,1)=DSMIVAL(CENSUSvalues,RNB,SEC(IND))
line 199:    & *UNCONFA
line 200: c For now we use a matrix with total load forecast for INDUSTRIAL sector
line 201: c When the INDUSTRIAL sector model is ready it should be replaced with
line 202: c a matrix with forecast for each end-use like: INDDM(year,reg,e-u)
line 203:  I=I+1
line 204:  ENDDO
line 205:  I=I-1
line 206:  IF(1,NE.EUINDEX(SEC(IND),2)) GOTO 999
line 207: C***** Termination of the Program/Subprogram *****
line 208:  RETURN
line 209: 999 WRITE(IMG,*)'<)) Message from subroutine DSMAIN'
line 210:  WRITE(IMG,*)'<)) Wrong number of end-uses specified on ',
line 211:  &'LDSMSTRU file for INDUSTRIAL sector, region: ',NERCnam(RNB)
line 212:  WHOOPS=.TRUE.
line 213:  RETURN
line 214:  END
line 215:  SUBROUTINE DSMARE(WHOOPS)
line 216: C***** Description of the Program/Subprogram *****
line 217: C This subroutine creates load forecasts for each end-use and NERC region
line 218: C using data imported from the RESIDENTIAL sector demand forecasting module.
line 219: C-----
line 220: C  Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 221: C*****
line 222:  IMPLICIT NONE
line 223: C***** Typing, Declaring and Initializing Constant Parameters ***
line 224:  INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 225:  INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 226:  INCLUDE 'NCNTRL' !<< global NEMS variables
line 227: C***** Typing and Declaring Variables *****
line 228:  REAL*4 load ! total load
line 229:  INTEGER*2 I,k ! universal counters
line 230:  REAL*4 HTRCON,COOLCN,H2OCON,REFCON,FRZCON,LTCON,APCON
line 231:  REAL*4 RSFLCN
line 232:  REAL*4 CKCON,DRYCON,SHTCON
line 233:  INTEGER*2 tgn ! technology group = end-use number
line 234:  LOGICAL WHOOPS ! error flag
line 235:  REAL*4 CENSUSvalues(MAXCRG) ! temporary vector with CENSUS division values
line 236:  REAL*4 FORESIGHTadj(MAXCRG) ! FORESIGHT adjustment factor
line 237:  EXTERNAL DSMIVAL
line 238: C***** COMMON and EQUIVALENCE *****
line 239:  INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 240:  INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 241:  INCLUDE 'DSMNERCR' !<< NERC region data
line 242:  INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 243:  INCLUDE 'QBLK' !<< supplies some constants required by MXQBLK
line 244:  INCLUDE 'MXQBLK' !<< foresight data
line 245:  COMMON /HTCN/ HTRCON(NYRRD,NEFHTR,NCRRD) ! HEATING

```



```

line 246: COMMON /CLCN/ COOLCN(NYRRD,NEFCOO,NCRRD) ! COOLING
line 247: COMMON /HWCN/ H2OCON(NYRRD,NEFWHR,NCRRD) ! W.HEAT.
line 248: COMMON /RFCN/ REFCON(NYRRD,NCRRD) ! REFRIG.
line 249: COMMON /FZCN/ FRZCON(NYRRD,NCRRD) ! FREEZERS
line 250: COMMON /LTC/ LTCN(NYRRD,NCRRD) ! LIGHTING
line 251: COMMON /APC/ APCON(NYRRD,NCRRD) ! APPLIENC.
line 252: COMMON /CKCN/ CKCON(NYRRD,NEFSTO,NCRRD) ! STOVES
line 253: COMMON /DRYCN/ DRYCON(NYRRD,NEFDRY,NCRRD)! DRYERS
line 254: COMMON /SHC/ SHTCON(NYRRD,NEFSHT,NCRRD) ! SECONDARY HEATING
line 255: C***** Initializing Variables *****
line 256: C***** Body of the Program/Subprogram *****
line 257: l=EUINDEX(SEC(RES),1)
line 258: tgn=0
line 259: C Calculate current year FORESIGHT adjustment factor
line 260: DO k=1,nCENSUSreg
line 261: IF(XQELRS(k,CURIYR).LE.0.0) GOTO 998
line 262: FORESIGHTadj(k)=XQELRS(k,K1)/XQELRS(k,CURIYR)
line 263: ENDDO
line 264: C Aggregate CENSUS regions load forecast into NERC region forecasts by end-use
line 265: C HEATING
line 266: DO k=1,nCENSUSreg
line 267: CENSUSvalues(k)=HTRCON(CURIYR,ELHEATX,k)*FORESIGHTadj(k)
line 268: ENDDO
line 269: LFinum=LFinum+1
line 270: LoadForec(LFinum,1)=DSMNVAL(CENSUSvalues,RNB,SEC(RES))*UNCONFAR
line 271: tgn=tgn+1
line 272: ResTGdem(tgn)=LoadForec(LFinum,1)
line 273: C COOLING
line 274: l=l+1
line 275: DO k=1,nCENSUSreg
line 276: CENSUSvalues(k)=COOLCN(CURIYR,ELCOOLX,k)*FORESIGHTadj(k)
line 277: ENDDO
line 278: LFinum=LFinum+1
line 279: LoadForec(LFinum,1)=DSMNVAL(CENSUSvalues,RNB,SEC(RES))*UNCONFAR
line 280: tgn=tgn+1
line 281: ResTGdem(tgn)=LoadForec(LFinum,1)
line 282: C WATER HEATING
line 283: l=l+1
line 284: DO k=1,nCENSUSreg
line 285: CENSUSvalues(k)=H2OCON(CURIYR,ELWHEAX,k)*FORESIGHTadj(k)
line 286: ENDDO
line 287: LFinum=LFinum+1
line 288: LoadForec(LFinum,1)=DSMNVAL(CENSUSvalues,RNB,SEC(RES))*UNCONFAR
line 289: tgn=tgn+1
line 290: ResTGdem(tgn)=LoadForec(LFinum,1)
line 291: C REFRIGERATORS
line 292: l=l+1
line 293: DO k=1,nCENSUSreg
line 294: CENSUSvalues(k)=REFCON(CURIYR,k)*FORESIGHTadj(k)
line 295: ENDDO
line 296: LFinum=LFinum+1
line 297: LoadForec(LFinum,1)=DSMNVAL(CENSUSvalues,RNB,SEC(RES))*UNCONFAR
line 298: tgn=tgn+1
line 299: ResTGdem(tgn)=LoadForec(LFinum,1)
line 300: C FREEZERS
line 301: l=l+1
line 302: DO k=1,nCENSUSreg
line 303: CENSUSvalues(k)=FRZCON(CURIYR,k)*FORESIGHTadj(k)
line 304: ENDDO
line 305: LFinum=LFinum+1
line 306: LoadForec(LFinum,1)=DSMNVAL(CENSUSvalues,RNB,SEC(RES))*UNCONFAR
line 307: tgn=tgn+1

```

```

line 308:   ResTGdem(tgn)=LoadForec(LFinum,1)
line 309: C LIGHTING
line 310:   l=l+1
line 311:   DO k=1,nCENSUSreg
line 312:     CENSUSvalues(k)=LTCN(CURIYR,k)*FORESIGHTadj(k)
line 313:   ENDDO
line 314:   LFinum=LFinum+1
line 315:   LoadForec(LFinum,1)=DSMNVAL(CENSUSvalues,RNB,SEC(RES))*UNCONFAR
line 316:   tgn=tgn+1
line 317:   ResTGdem(tgn)=LoadForec(LFinum,1)
line 318: C applianceS
line 319:   l=l+1
line 320:   DO k=1,nCENSUSreg
line 321:     CENSUSvalues(k)=APCON(CURIYR,k)*FORESIGHTadj(k)
line 322:   ENDDO
line 323:   LFinum=LFinum+1
line 324:   LoadForec(LFinum,1)=DSMNVAL(CENSUSvalues,RNB,SEC(RES))*UNCONFAR
line 325:   tgn=tgn+1
line 326:   ResTGdem(tgn)=LoadForec(LFinum,1)
line 327: C STOVES
line 328:   l=l+1
line 329:   DO k=1,nCENSUSreg
line 330:     CENSUSvalues(k)=CKCON(CURIYR,ELSTOVX,k)*FORESIGHTadj(k)
line 331:   ENDDO
line 332:   LFinum=LFinum+1
line 333:   LoadForec(LFinum,1)=DSMNVAL(CENSUSvalues,RNB,SEC(RES))*UNCONFAR
line 334:   tgn=tgn+1
line 335:   ResTGdem(tgn)=LoadForec(LFinum,1)
line 336: C DRYERS
line 337:   l=l+1
line 338:   DO k=1,nCENSUSreg
line 339:     CENSUSvalues(k)=DRYCON(CURIYR,ELSTOVX,k)*FORESIGHTadj(k)
line 340:   ENDDO
line 341:   LFinum=LFinum+1
line 342:   LoadForec(LFinum,1)=DSMNVAL(CENSUSvalues,RNB,SEC(RES))*UNCONFAR
line 343:   tgn=tgn+1
line 344:   ResTGdem(tgn)=LoadForec(LFinum,1)
line 345: C SECONDARY HEATING
line 346:   l=l+1
line 347:   DO k=1,nCENSUSreg
line 348:     CENSUSvalues(k)=CKCON(CURIYR,ELSECHX,k)*FORESIGHTadj(k)
line 349:   ENDDO
line 350:   LFinum=LFinum+1
line 351:   LoadForec(LFinum,1)=DSMNVAL(CENSUSvalues,RNB,SEC(RES))*UNCONFAR
line 352:   tgn=tgn+1
line 353:   ResTGdem(tgn)=LoadForec(LFinum,1)
line 354:   IF(1.NE.EUINDEX(SEC(RES),2)) GOTO 999
line 355: C***** Termination of the Program/Subprogram *****
line 356:   RETURN
line 357: 999 WRITE(IMG,*)'<)) Message from subroutine DSMARE'
line 358:   WRITE(IMG,*)'<)) Wrong number of end-uses specified on ',
line 359:   &'LDSMSTRU file for RESIDENTIAL sector, region: ',NERCnam(RNB)
line 360:   WHOOPS=.FALSE.
line 361:   RETURN
line 362: 998 WRITE(IMG,*)'<)) Message from subroutine DSMARE'
line 363:   WRITE(IMG,*)'<)) Foresight residential demand for year',CURIYR
line 364:   &,' CENSUS region #',k,' equal to 0.0'
line 365:   WHOOPS=.FALSE.
line 366:   RETURN
line 367:   END
line 368:   SUBROUTINE DSMATR(WHOOPS)
line 369: C***** Description of the Program/Subprogram *****

```

```

line 370: C This subroutine creates load forecasts for each end-use and NERC region
line 371: C using data imported from the TRANSPORT. sector demand forecasting module.
line 372: C-----
line 373: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 374: C*****
line 375:   IMPLICIT NONE
line 376: C***** Typing, Declaring and Initializing Constant Parameters ***
line 377:   INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 378:   INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 379:   INCLUDE 'NCNTRL' !<< global NEMS variables
line 380: C***** Typing and Declaring Variables *****
line 381:   REAL*4 load ! total load
line 382:   INTEGER*2 k,l ! universal counters
line 383:   LOGICAL WHOOPS ! error flag
line 384:   REAL*4 CENSUSvalues(MAXCRG) ! temporary vector with CENSUS division values
line 385:   REAL*4 FORESIGHTadj(MAXCRG) ! FORESIGHT adjustment factor
line 386:   EXTERNAL DSMNVAL
line 387: C***** COMMON and EQUIVALENCE *****
line 388:   INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 389:   INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 390:   INCLUDE 'DSMNERCR' !<< NERC region data
line 391:   INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 392:   INCLUDE 'QBLK' !<< supplies some constants required by MXQBLK
line 393:   INCLUDE 'MXQBLK' !<< foresight data
line 394:   INCLUDE 'TRANREP' !<< access to transportation sector demand
line 395: C***** Initializing Variables *****
line 396: C***** Body of the Program/Subprogram *****
line 397:   l=EUINDEX(SEC(TRA),1)
line 398:   DO k=1,nCENSUSreg
line 399:     FORESIGHTadj(k)=XQELTR(k,K1)/XQELTR(k,CURIYR)
line 400:   ENDDO
line 401: C Aggregate CENSUS regions load forecast into NERC region forecasts by end-use
line 402: C LIGHT DUTY ELECTRIC VEHICLES
line 403:   DO k=1,nCENSUSreg
line 404:     CENSUSvalues(k)=TRQLDV(ELLDVHX,k,CURIYR)*FORESIGHTadj(k)
line 405:   ENDDO
line 406:   LFinum=LFinum+1
line 407:   LoadForec(LFinum,1)=DSMNVAL(CENSUSvalues,RNB,SEC(TRA))
line 408:   & *UNCONFA
line 409: C ELECTRIC TRAINS
line 410:   l=l+1
line 411:   DO k=1,nCENSUSreg
line 412:     CENSUSvalues(k)=TRQRAILR(ELRAILX,k,CURIYR)*FORESIGHTadj(k)
line 413:   ENDDO
line 414:   LFinum=LFinum+1
line 415:   LoadForec(LFinum,1)=DSMNVAL(CENSUSvalues,RNB,SEC(TRA))
line 416:   & *UNCONFA
line 417:   IF(1.NE.EUINDEX(SEC(TRA),2)) GOTO 999
line 418: C***** Termination of the Program/Subprogram *****
line 419:   RETURN
line 420: 999 WRITE(IMG,*)'<)) Message from subroutine DSMATR'
line 421:   WRITE(IMG,*)'<)) Wrong number of end-uses specified on ',
line 422:   &'LDSMSTRU file for TRANSPORT. sector, region: ',NERCnam(RNB)
line 423:   WHOOPS=.TRUE.
line 424:   RETURN
line 425:   END
line 426:   SUBROUTINE DSMCADJ
line 427: C***** Description of the Program/Subprogram *****
line 428: C This subroutine calculates adjustment factors for COMMERCIAL sector load
line 429: C so the by-end-use demand match the totals for the sector passed
line 430: C by commercial demand forecasting module to NEMS
line 431: C-----

```

```

line 432: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 433: C*****
line 434:   IMPLICIT NONE
line 435: C***** Typing, Declaring and Initializing Constant Parameters ****
line 436:   INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 437:   INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 438:   INCLUDE 'NCNTRL' !<< global NEMS variables
line 439: C***** Typing and Declaring Variables *****
line 440:   REAL*4 load ! total load
line 441:   INTEGER*2 l,i,j,k ! universal counters
line 442:   LOGICAL WHOOPS ! error flag
line 443: C***** COMMON and EQUIVALENCE *****
line 444:   INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 445:   INCLUDE 'COMPARM' !<< defines PARAMETERS: CMnumBldg,CMnumServ
line 446:   INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 447:   INCLUDE 'DSMNERCR' !<< NERC region data
line 448:   INCLUDE 'QBLK' !<< supplies some constants needed in MXQBLK
line 449: C***** Initializing Variables *****
line 450: C***** Body of the Program/Subprogram *****
line 451:   DO k=1,nCENSUSreg
line 452:     load=0.0
line 453:     DO i=1,CMnumBldg
line 454:       DO j=1,CMnumServ
line 455:         load=load+EndUseConsump(ELINDEX,j,i,k,CURIYR)
line 456:       ENDDO
line 457:     ENDDO
line 458:     IF(load.GT.0.0) THEN
line 459:       ComDemAdjFac(k)=QELCM(k,CURIYR)/load
line 460:     ELSE
line 461:       ComDemAdjFac(k)=0.0
line 462:     ENDIF
line 463:   ENDDO
line 464: C***** Termination of the Program/Subprogram *****
line 465:   RETURN
line 466:   END
line 467:   LOGICAL FUNCTION DSMCMP(STR1,STR2)
line 468: C***** Description of the Program/Subprogram *****
line 469: C This function compares two strings. If any of the two strings is included
line 470: C in the other, the value is .TRUE., otherwise .FALSE.
line 471: C-----
line 472: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 473: C*****
line 474:   IMPLICIT NONE
line 475: C***** Typing, Declaring and Initializing Constant Parameters ****
line 476: C***** Typing and Declaring Variables *****
line 477:   CHARACTER*(*) STR1 ! string 1
line 478:   CHARACTER*(*) STR2 ! string 2
line 479:   INTEGER*2 l1 ! length of string1
line 480:   INTEGER*2 l2 ! length of string2
line 481:   INTEGER*2 b1 ! first blank position in string1
line 482:   INTEGER*2 b2 ! first blank position in string2
line 483:   INTEGER*2 I ! temporary variable
line 484: C***** COMMON and EQUIVALENCE *****
line 485: C***** Initializing Variables *****
line 486:   DSMCMP=.TRUE.
line 487: C***** Body of the Program/Subprogram *****
line 488:   DSMCMP=.FALSE.
line 489:   l1=LEN(STR1)
line 490:   l2=LEN(STR2)
line 491:   b1=INDEX(STR1,' ')
line 492:   b2=INDEX(STR2,' ')
line 493:   IF(b1.GT.0) THEN

```

```

line 494:    IF(b2.GT.0) THEN
line 495:    IF(b1.LT.b2) THEN
line 496:        DO I=b1,I1
line 497:            STR1(I:I)=' '
line 498:        ENDDO
line 499:        DO I=b1,I2
line 500:            STR2(I:I)=' '
line 501:        ENDDO
line 502:    ELSE
line 503:        DO I=b2,I1
line 504:            STR1(I:I)=' '
line 505:        ENDDO
line 506:        DO I=b2,I2
line 507:            STR2(I:I)=' '
line 508:        ENDDO
line 509:    ENDIF
line 510: ELSE
line 511:     DO I=b1,I1
line 512:         STR1(I:I)=' '
line 513:     ENDDO
line 514:     DO I=b1,I2
line 515:         STR2(I:I)=' '
line 516:     ENDDO
line 517: ENDIF
line 518: ELSE IF(b2.GT.0) THEN
line 519:     DO I=b2,I1
line 520:         STR1(I:I)=' '
line 521:     ENDDO
line 522:     DO I=b2,I2
line 523:         STR2(I:I)=' '
line 524:     ENDDO
line 525: ENDIF
line 526: IF(11.GT.12) THEN
line 527:     IF(INDEX(STR1,STR2).EQ.1) DSMCMP=.TRUE.
line 528: ELSE
line 529:     IF(INDEX(STR2,STR1).EQ.1) DSMCMP=.TRUE.
line 530: ENDIF
line 531: C***** Termination of the Program/Subprogram *****
line 532:     RETURN
line 533: END
line 534: REAL*4 FUNCTION DSMCVAL(NERCvalues,CENSUSdiv,sector)
line 535: C***** Description of the Program/Subprogram *****
line 536: C This routine develops a CENSUS div. value based on the values for the NERC
line 537: C regions. It uses different mapping matrices for different demand sectors.
line 538: C-----
line 539: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 540: C*****
line 541:     IMPLICIT NONE
line 542: C***** Typing, Declaring and Initializing Constant Parameters ****
line 543:     INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 544:     INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 545:     INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 546:     INCLUDE 'DSMNERCR' !<< NERC region data
line 547: C***** Typing and Declaring Variables *****
line 548:     REAL*4 NERCvalues(*)      ! CENSUS division values
line 549:     INTEGER*2 CENSUSdiv      ! current NERC region
line 550:     INTEGER*2 sector         ! current sector
line 551:     INTEGER*2 k              ! universal counter
line 552: C***** COMMON and EQUIVALENCE *****
line 553: C***** Initializing Variables *****
line 554: C***** Body of the Program/Subprogram *****
line 555:     DSMCVAL=0.0

```

```

line 556: DO k=1,nNERCreg
line 557:   DSMCVAL=DSMCVAL+NERCvalues(k)*MappNtoC(k,CENSUSdiv,sector)
line 558: ENDDO
line 559: C***** Termination of the Program/Subprogram *****
line 560: RETURN
line 561: END
line 562: SUBROUTINE DSMDLT
line 563: C***** Description of the Program/Subprogram *****
line 564: C This subroutine calculates load forecast for so-called DELTA approach
line 565: C-----
line 566: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 567: C*****
line 568:   IMPLICIT NONE
line 569: C***** Typing, Declaring and Initializing Constant Parameters ***
line 570:   INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 571:   INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 572: C***** Typing and Declaring Variables *****
line 573:   INTEGER*2 I ! temporary variable
line 574: C   REAL*4 slr ! system load ratio
line 575: C***** COMMON and EQUIVALENCE *****
line 576:   INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 577:   INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 578: C***** Initializing Variables *****
line 579: C***** Body of the Program/Subprogram *****
line 580: C***** Original delta approach *****
line 581: C   SysLoad=0.0
line 582: C   DO I=1,LFinum
line 583: C     SystemLoad=SysLoad+LoadForec(I,1)
line 584: C   ENDDO
line 585: C   slr=SystemLoad/BaseYrSysLd(RNB)
line 586: C   DO I=1,LFinum
line 587: C     LoadForec(I,2)=LoadForec(I,1)-slr*BaseYrLd(I,RNB)
line 588: C   ENDDO
line 589: C*****
line 590: C* Approach where all load above the base year usage is modelled with e-u's LSR
line 591:   DO I=1,LFinum
line 592:     LoadForec(I,2)=LoadForec(I,1)-BaseYrLd(I,RNB)
line 593:   ENDDO
line 594:   SystemLoad=BaseYrSysLd(RNB)
line 595: C***** Termination of the Program/Subprogram *****
line 596: RETURN
line 597: END
line 598: SUBROUTINE DSMECP1(WHOOPS)
line 599: C***** Description of the Program/Subprogram *****
line 600: C This routine prepares data for the foresight years for ECP
line 601: C-----
line 602: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 603: C*****
line 604:   IMPLICIT NONE
line 605: C***** Typing, Declaring and Initializing Constant Parameters ***
line 606:   INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 607:   INCLUDE 'NCNTRL' !<< global NEMS variables
line 608:   INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 609:   INCLUDE 'EMM$PARM' !<< EMM GLOBAL PARAMETERS
line 610: C***** Typing and Declaring Variables *****
line 611:   LOGICAL WHOOPS ! error flag
line 612: C***** COMMON and EQUIVALENCE *****
line 613:   INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 614:   INCLUDE 'DSMNEMSC' !<< results of LDSM to be passed to the rest of NEMS
line 615:   INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 616: C***** Initializing Variables *****
line 617: C***** Body of the Program/Subprogram *****

```

```

line 618: ECPLastYearIndex=CURIYR+ECP$XPH-1 ! Calculate last year index for ECP run
line 619: CALL DSMLCP(1) ! Develop LDC for ECP for CURIYR
line 620: DO K1=CURIYR+1,ECPLastYearIndex ! do for all future years of ECP horizon
line 621:     ! Because K1 is a parameter of a global type, used by
line 622:     ! many levels of routines within the loop, its value
line 623:     ! is passed through a common block, rather than as
line 624:     ! explicit subroutine parameter
line 625:     CALL DSMFOR ! Prepare load forecast
line 626:     CALL DSMHLM ! Run procedures which develop system load
line 627:     CALL DSMLCP(1) ! Develop LDC for ECP
line 628:     IF(CURIYR.GT.FIRSYR) THEN
line 629:         IF(K1.EQ.CURIYR+1) THEN
line 630:             CALL DSMPRGD ! Initially screen DSM options in DSM programs
line 631:             CALL DSMPCIM(WHOOPS) ! Determine DSM programs' costs and impacts
line 632:             IF (WHOOPS) RETURN
line 633:         ENDIF
line 634:         CALL DSMPRGL ! Develop DSM Program Load Duration Curves
line 635:     ENDIF
line 636: ENDDO
line 637: C***** Termination of the Program/Subprogram *****
line 638:     RETURN
line 639:     END
line 640: SUBROUTINE DSMECP2
line 641: C***** Description of the Program/Subprogram *****
line 642: C This routine reads ECP choices of the DSM programs and translates them
line 643: C to the information required by the demand forecasting modules
line 644: C-----
line 645: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 646: C*****
line 647:     IMPLICIT NONE
line 648: C***** Typing, Declaring and Initializing Constant Parameters ****
line 649:     INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 650:     INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 651:     INCLUDE 'EMM$PARM' !<< EMM GLOBAL PARAMETERS
line 652:     INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 653: C***** Typing and Declaring Variables *****
line 654:     INTEGER*2 I,J,I1,k,n,nopt,c,YY
line 655:     INTEGER*2 CYR1
line 656:     INTEGER*2 opti
line 657:     REAL*4 delta,ppramp
line 658:     REAL*4 Rshift(MAXNRG,MAXRDSMO),Cshift(MAXNRG,MAXCDSMO)
line 659:     REAL*4 NERCvalues(MAXNRG)
line 660:     EXTERNAL DSMCVAL
line 661: C***** COMMON and EQUIVALENCE *****
line 662:     INCLUDE 'NCNTRL' !<< global NEMS variables
line 663:     INCLUDE 'DSMTFECF' !<< communication with ECP
line 664:     INCLUDE 'DSMPROG' !<< DSM program data
line 665:     INCLUDE 'DSMOPTDB' !<< DSM options data base
line 666:     INCLUDE 'DSMNERCR' !<< NERC region data
line 667:     INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 668:     INCLUDE 'DSMTFCOM' !<< variables for communication with COMMERCIAL model
line 669:     INCLUDE 'DSMTFRES' !<< communication with RESIDENTIAL model
line 670: C***** Initializing Variables *****
line 671:     DO k=1,MAXNRG
line 672:         DO I=1,MAXCDSMO
line 673:             RopChoice(CURIYR,k,I)=0.0 ! These are choices made in CURIYR
line 674:         ENDDO ! but they always apply to CURIYR+1
line 675:         DO I=1,MAXRDSMO ! In the FIRSYR ECP does not chose
line 676:             CopChoice(CURIYR,k,I)=0.0 ! options at all
line 677:         ENDDO
line 678:     ENDDO
line 679:     CYR1=CURIYR-1

```

```

line 680: C***** Body of the Program/Subprogram *****
line 681: C Calculate fractions of the markets available for options that have been
line 682: C decided by ECP in the current iteration to be affected by the options
line 683:   DO k=1,nNERCreg ! Do for each NERC region to be processed
line 684: C FOR RESIDENTIAL
line 685: C Calculate fractions of the market already committed to the options, because of
line 686: C the decisions from the past
line 687:   DO opti=1,MAXRDSMO
line 688:     Rshift(k,opti)=0.0
line 689:     DO YY=FIRSYR+1,CYR1
line 690: C       ppramp=1.0/DSMROptionRamp(opti)*(CURIYR-YY)
line 691: C       IF(ppramp.GT.1.0) ppramp=1.0
line 692:         Rshift(k,opti)=Rshift(k,opti)+RopChoice(YY,k,opti)
line 693:       ENDDO
line 694:     ENDDO
line 695:     DO I=1,NRPROG
line 696:       DO J=1,RPROGDEFN(k,I)
line 697:         IF(RPROGDEF(k,I,J,2).EQ.6) THEN
line 698:           RopChoice(CURIYR,k,RPROGDEF(k,I,J,1))=
line 699: &     DSMPRCHOICE(k,I)*(1.0-Rshift(k,RPROGDEF(k,I,J,1)))
line 700:         ENDIF
line 701:       ENDDO
line 702:     ENDDO
line 703: C FOR COMMERCIAL
line 704: C Calculate fractions of the market already committed to the options, because of
line 705: C the decisions from the past
line 706:   DO opti=1,MAXCDSMO
line 707:     Cshift(k,opti)=0.0
line 708:     DO YY=FIRSYR+1,CYR1
line 709: C       ppramp=1.0/DSMCOptionRamp(opti)*(CURIYR-YY)
line 710: C       IF(ppramp.GT.1.0) ppramp=1.0
line 711:         Cshift(k,opti)=Cshift(k,opti)+RopChoice(YY,k,opti)
line 712:       ENDDO
line 713:     ENDDO
line 714:     DO I=1,NCPROG
line 715:       I1=I+NRPROG
line 716:       DO J=1,CPROGDEFN(k,I)
line 717:         IF(CPROGDEF(k,I,J,2).EQ.6) THEN
line 718:           CopChoice(CURIYR,k,CPROGDEF(k,I,J,1))=
line 719: &     DSMPRCHOICE(k,I1)*(1.0-Cshift(k,CPROGDEF(k,I,J,1)))
line 720:         ENDIF
line 721:       ENDDO
line 722:     ENDDO
line 723:   ENDDO
line 724: C Calculate cumulative fractions of the markets available for options that have
line 725: C been decided until now by ECP to be affected by the options
line 726: C FOR RESIDENTIAL
line 727:   n=0
line 728:   DO c=1,nCENSUSreg
line 729:     nopt=0
line 730:     DO J=1,MAXRDSMO
line 731:       DO k=1,nNERCreg
line 732:         NERCvalues(k)=RopChoice(CURIYR,k,J)+Rshift(k,J)
line 733:       ENDDO
line 734:       delta=DSMCVAL(NERCvalues,c,SEC(RES))
line 735:       IF(delta.GT.0) THEN
line 736:         nopt=nopt+1
line 737:         n=n+1
line 738:         DSMrOptionIndex(n)=J
line 739:         DSMrFracOptionMarket(n)=delta
line 740:       ENDIF
line 741:     ENDDO

```



```

line 742:   DSMrOptionsNumber(c)=nopt
line 743:   ENDDO
line 744: C FOR COMMERCIAL
line 745:   n=0
line 746:   DO c=1,nCENSUSreg
line 747:     nopt=0
line 748:     DO J=1,MAXCDSMO
line 749:       DO k=1,nNERCreg
line 750:         NERCvalues(k)=CopChoice(CURIYR,k,J)+Cshift(k,J)
line 751:       ENDDO
line 752:       delta=DSMCVAL(NERCvalues,c,SEC(COM))
line 753:       IF(delta.GT.0) THEN
line 754:         nopt=nopt+1
line 755:         n=n+1
line 756:         DSMcOptionIndex(n)=J
line 757:         DSMcFracOptionMarket(n)=delta
line 758:       ENDIF
line 759:     ENDDO
line 760:   DSMcOptionsNumber(c)=nopt
line 761:   ENDDO
line 762: C***** Termination of the Program/Subprogram *****
line 763:   RETURN
line 764:   END
line 765:   SUBROUTINE DSMEFD
line 766: C***** Description of the Program/Subprogram *****
line 767: C   This program develops LDC's for EFD module
line 768: C-----
line 769: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 770: C*****
line 771:   IMPLICIT NONE
line 772: C***** Typing, Declaring and Initializing Constant Parameters ****
line 773:   INTEGER*2 ONE
line 774:   PARAMETER(ONE=1)
line 775:   INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 776:   INCLUDE 'NCNTRL' !<< global NEMS variables
line 777:   INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 778: C***** Typing and Declaring Variables *****
line 779:   REAL*4 SYLOAD1(MAXRHOUR) !copy of SYLOAD for one segment of hour # in year
line 780:   REAL*4 SYLOAD2(MAXRHOUR) !same as SYLOAD1 but different order
line 781:   INTEGER*2 nBlockInSeason ! index of current block in season
line 782:   INTEGER*2 nBlockInLDC ! index of current block in LDC
line 783:   INTEGER*2 iseg ! segment index in LDC
line 784:   INTEGER*2 isegment ! segment index in a season
line 785:   INTEGER*2 iseason ! season index
line 786:   INTEGER*2 h,hn,i,nb,j,n
line 787:   INTEGER*2 ih1,ih2 !left/right-hand integer number of hours for hCoordinate
line 788:   REAL*4 yCoordinate(MAXEFDB) ! load values for each EFD LDC data point
line 789:   INTEGER*2 BlockRank(MAXEFDB) ! data points ranks in segments
line 790:   INTEGER*2 SegAssign(MAXEFDB) ! data point assignments of the blocks
line 791:   INTEGER*2 BlockInd(MAXEFDB) ! indices of data points while sorted in segm.
line 792:   REAL*4 peak, knee ! peak,knee load in season
line 793:   INTEGER*2 ipeak,iknee ! segment assignment of peak and knee
line 794: C***** COMMON and EQUIVALENCE *****
line 795:   INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 796:   INCLUDE 'DSMTOEFD' !<< communication with EFD
line 797:   INCLUDE 'DSMNEMSC' !<< results of LDSM to be passed to the rest of NEMS
line 798:   INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 799:   INCLUDE 'DSMCALDR' !<< calendar data
line 800: C***** Initializing Variables *****
line 801: C***** Body of the Program/Subprogram *****
line 802: C
line 803: C Sort loads in SYLOAD

```

```

line 804: C Create a vector of hourly loads for all calendar hours
line 805:   DO hn=1,nhour
line 806:     SYLOAD1(hn)=SYLOAD(hn)
line 807:   ENDDO
line 808: C Rewrite SYLOAD1 to SYLOAD2 in segment order
line 809:   DO i=1,nhour
line 810:     SYLOAD2(i)=SYLOAD1(HourNumberEFD(i))
line 811:   ENDDO
line 812: C Sort loads in each segment in descending order
line 813:   DO iseg=1,numSEFD
line 814:     CALL DSMQSR(SYLOAD2,HourNumberEFD,
line 815:       &      nFirstCalHinSeg(iseg),nLastCalHinSeg(iseg))
line 816:   ENDDO
line 817: C Expand calendar year loads into real year loads
line 818:   h=0
line 819:   DO hn=1,nhour
line 820:     DO i=1,HourlyWeights(HourNumberEFD(hn))
line 821:       h=h+1
line 822:       SYLOAD1(h)=SYLOAD2(hn)
line 823:     ENDDO
line 824:   ENDDO
line 825: C Calculate heights of the blocks
line 826:   iseg=numSEFD+1
line 827:   nBlockInLDC=0
line 828:   DO  iseason=EFDnS,1,-1
line 829:     nBlockInSeason=0
line 830:     peak=0.0
line 831:     knee=1.0E+30
line 832:     DO  isegment=EFDnumSeg(iseason),1,-1
line 833:       iseg=iseg-1
line 834:       IF(SYLOAD1(nFirstHinSeg(iseg)).GT.peak) THEN
line 835:         peak=SYLOAD1(nFirstHinSeg(iseg))
line 836:         ippeak=isegment
line 837:       ENDIF
line 838:       IF(SYLOAD1(nLastHinSeg(iseg)).LT.knee) THEN
line 839:         knee=SYLOAD1(nLastHinSeg(iseg))
line 840:         iknee=isegment
line 841:       ENDIF
line 842:       DO i=1,NumberOfBlocksinSeg(iseg)
line 843:         nBlockInLDC=nBlockInLDC+1
line 844:         nBlockInSeason=nBlockInSeason+1
line 845:         ih1=hCoordinate(nBlockInLDC)
line 846:         ih2=hCoordinate(nBlockInLDC)+0.5
line 847:         yCoordinate(nBlockInSeason)=(SYLOAD1(ih1)-SYLOAD1(ih2))*
line 848:         &   (ih2-hCoordinate(nBlockInLDC))+SYLOAD1(ih2)
line 849:         BlockInd(nBlockInSeason)=nBlockInSeason
line 850:         BlockRank(nBlockInSeason)=i
line 851:         SegAssign(nBlockInSeason)=isegment
line 852:       ENDDO
line 853:     ENDDO
line 854: C Sort blocks in each of the seasons in descending order to produce LDC
line 855:   CALL DSMQSR(yCoordinate,BlockInd,ONE,nBlockInSeason)
line 856:   DO nb=1,nBlockInSeason
line 857:     n=nb+1
line 858:     EFDLDCYC(RNB,iseason,n)=yCoordinate(nb)
line 859:     EFDLDCPR(RNB,iseason,n)=BlockRank(BlockInd(nb))
line 860:     EFDLDCSA(RNB,iseason,n)=SegAssign(BlockInd(nb))
line 861:   ENDDO
line 862:   nBlockInSeason=nBlockInSeason+2
line 863:   EFDLDCYC(RNB,iseason,1)=peak
line 864:   EFDLDCPR(RNB,iseason,1)=0
line 865:   EFDLDCSA(RNB,iseason,1)=ipeak

```

```

line 866:   EFDLDCYC(RNB,iseason,nBlockInSeason)=knee
line 867:   EFDLDCPR(RNB,iseason,nBlockInSeason)=0
line 868:   EFDLDCSA(RNB,iseason,nBlockInSeason)=iknee
line 869:   ENDDO
line 870: C***** Termination of the Program/Subprogram *****
line 871:   RETURN
line 872:   END
line 873:   SUBROUTINE DSMEFP
line 874: C***** Description of the Program/Subprogram *****
line 875: C This subroutine finds sectorial peaks required by EFP
line 876: C-----
line 877: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 878: C*****
line 879:   IMPLICIT NONE
line 880: C***** Typing, Declaring and Initializing Constant Parameters ****
line 881:   INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 882:   INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 883: C***** Typing and Declaring Variables *****
line 884:   INTEGER*2 I,J,K,L,M,N ! multi-purpose pointers
line 885:   REAL*4 peak ! current maximum value of load
line 886:   REAL*4 average ! temporary variable for averaging
line 887:   INTEGER*2 phour ! current hour of peak
line 888:   INTEGER*2 pday ! current peak-day
line 889: C***** COMMON and EQUIVALENCE *****
line 890:   INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 891:   INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 892:   INCLUDE 'DSMTFEFP' !<< communication with EFP
line 893:   INCLUDE 'DSMCALDR' !<< calendar data
line 894:   INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 895: C***** Initializing Variables *****
line 896: C*** Find annual COINCIDENT/NON-COINCIDENT peaks for sectors
line 897:   DO I=1,NumSec
line 898:     SecAnnualPeak(RNB,I,1)=SectorLoad(SystemPeakHour(K1),I)
line 899: C CALCULATE AVERAGE OF NpeakH COINCIDENT SECTORIAL PEAK HOURS
line 900:   average=0.0
line 901:   DO K=1,NpeakH
line 902:     average=average+SectorLoad(SysPeakHour(K),I)
line 903:   ENDDO
line 904:   average=average/NpeakH
line 905:   SecAnnPeaAvPCP(RNB,I)=average
line 906:   peak=0.0
line 907:   DO K=1,nhour
line 908:     IF(SectorLoad(K,I).GT.peak) THEN
line 909:       peak=SectorLoad(K,I)
line 910:     ENDIF
line 911:   ENDDO
line 912:   SecAnnualPeak(RNB,I,2)=peak
line 913: ENDDO
line 914:   SystemLF(RNB)=SystemLoadFactor(K1)
line 915: C***** Termination of the Program/Subprogram *****
line 916:   RETURN
line 917:   END
line 918:   SUBROUTINE DSMEFP2
line 919: C***** Description of the Program/Subprogram *****
line 920: C This subroutine calculates current year DSM costs for the needs of EFP
line 921: C-----
line 922: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 923: C*****
line 924:   IMPLICIT NONE
line 925: C***** Typing, Declaring and Initializing Constant Parameters ****
line 926:   INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 927:   INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations

```

```

line 928: C***** Typing and Declaring Variables *****
line 929:   INTEGER*2 I,J,K,L,M,N ! multi-purpose pointers
line 930:   INTEGER*2 opti ! index of DSM option
line 931:   REAL*4 REBATE ! rebates
line 932:   REAL*4 AnnualEnSavings ! AnnualEnSavings
line 933:   REAL*4 CENSUSvalues(MAXCRG)
line 934:   REAL*4 CV(MAXCRG,MAXRDSMO)
line 935:   EXTERNAL DSMNVAL
line 936: C***** COMMON and EQUIVALENCE *****
line 937:   INCLUDE 'NCNTRL' !<< global NEMS variables
line 938:   INCLUDE 'DSMTFEFP' !<< communication with EFP
line 939:   INCLUDE 'DSMTFRES' !<< communication with RESIDENTIAL model
line 940:   INCLUDE 'DSMTFCOM' !<< variables for communication with COMMERCIAL model
line 941:   INCLUDE 'DSMOPTDB' !<< DSM options data base
line 942:   INCLUDE 'DSMNERCR' !<< NERC region data
line 943:   INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 944: C***** Initializing Variables *****
line 945:   DO I=1,nNERCreg
line 946:     DO J=1,MAXSEC
line 947:       DSMAnnualCost(I,J)=0.0
line 948:     ENDDO
line 949:   ENDDO
line 950:   DO I=1,nCENSUSreg
line 951:     DO J=1,MAXRDSMO
line 952:       CV(I,J)=0.0
line 953:     ENDDO
line 954:   ENDDO
line 955: C***** Body of the Program/Subprogram *****
line 956: C   CALCULATE DSM COSTS FOR THE NEEDS OF EFP
line 957: C   FOR RESIDENTIAL SECTOR
line 958:   N=0
line 959:   DO I=1,nCENSUSreg
line 960:     DO J=1,DSMrOptionsNumber(I)
line 961:       N=N+1
line 962:       CV(I,DSMrOptionIndex(N))=DSMrNumUnitChange(N)
line 963:     ENDDO
line 964:   ENDDO
line 965:   DO opti=1,MAXRDSMO
line 966:     DO I=1,nCENSUSreg
line 967:       CENSUSvalues(I)=CV(I,opti)
line 968:     ENDDO
line 969:     DO K=1,nNERCreg
line 970:       CALL DSMREBR(opti,K,REBATE,AnnualEnSavings)
line 971:       DSMAnnualCost(K,SEC(RES))=DSMAnnualCost(K,SEC(RES))+
line 972:       & DSMNVAL(CENSUSvalues,K,SEC(RES))*(REBATE+AnnualEnSavings)
line 973:       & *DSMROptionCost(opti)
line 974:     ENDDO
line 975:   ENDDO
line 976: C   FOR COMMERCIAL SECTOR
line 977:   N=0
line 978:   DO I=1,nCENSUSreg
line 979:     DO J=1,DSMcOptionsNumber(I)
line 980:       N=N+1
line 981:       CV(I,DSMcOptionIndex(N))=DSMcServDemChange(N)
line 982:     ENDDO
line 983:   ENDDO
line 984:   DO opti=1,MAXRDSMO
line 985:     DO I=1,nCENSUSreg
line 986:       CENSUSvalues(I)=CV(I,opti)
line 987:     ENDDO
line 988:     DO K=1,nNERCreg
line 989:       CALL DSMREBC(opti,K,REBATE,AnnualEnSavings)

```

```

line 990:      DSMAnnualCost(K,SEC(COM))=DSMAnnualCost(K,SEC(COM))+
line 991:      & DSMNVAL(CENSUSvalues,K,SEC(COM))*(REBATE+AnnualEnSavings
line 992:      & *DSMCOptionCost(opti))
line 993:      ENDDO
line 994:      ENDDO
line 995: C***** Termination of the Program/Subprogram *****
line 996:      RETURN
line 997:      END
line 998:      SUBROUTINE DSMFOR(WHOOPS)
line 999: C***** Description of the Program/Subprogram *****
line 1000: C This subroutine prepares an end-use level load forecast for a current NERC
line 1001: C region based on existing CENSUS region load forecasts supplied by the demand
line 1002: C modules. Two versions of the load forecast by LSR are developed: one for the
line 1003: C delta approach, second for the traditional approach (used in computations of
line 1004: C monthly coincident/noncoincident sectoral peak loads).
line 1005: C-----
line 1006: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 1007: C*****
line 1008:      IMPLICIT NONE
line 1009: C***** Typing, Declaring and Initializing Constant Parameters ****
line 1010:      INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 1011:      INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 1012: C***** Typing and Declaring Variables *****
line 1013:      INTEGER*2 I ! current number of sector
line 1014:      REAL*4   load ! temporary variable to calculate total system load
line 1015:      LOGICAL*1 flag ! flag
line 1016:      LOGICAL WHOOPS ! error flag
line 1017: C***** COMMON and EQUIVALENCE *****
line 1018:      INCLUDE 'NCNTRL' !<< global NEMS variables
line 1019:      INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 1020:      INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 1021:      INCLUDE 'DSMHLM' !<< HELM algorithm variables
line 1022: C***** Initializing Variables *****
line 1023: C***** Body of the Program/Subprogram *****
line 1024: C Import the load forecast data from the other modules of NEMS and
line 1025: C translate them from CENSUS regions into the NERC regions
line 1026:      LFinum=0 ! index on the Load Forecast list
line 1027:      TotDemCensus=0.0
line 1028:      DO I=1,NumSec
line 1029:          IF (SEC(RES) .EQ. I) THEN
line 1030:              CALL DSMARE(WHOOPS)
line 1031:              IF(WHOOPS) RETURN
line 1032:          ELSE IF (SEC(COM) .EQ. I) THEN
line 1033:              CALL DSMACM(WHOOPS)
line 1034:              IF(WHOOPS) RETURN
line 1035:          ELSE IF (SEC(IND) .EQ. I) THEN
line 1036:              CALL DSMMAIN(WHOOPS)
line 1037:              IF(WHOOPS) RETURN
line 1038:          ELSE IF (SEC(TRA) .EQ. I) THEN
line 1039:              CALL DSMATR(WHOOPS)
line 1040:              IF(WHOOPS) RETURN
line 1041:          ELSE
line 1042:              GOTO 999
line 1043:          ENDIF
line 1044:      ENDDO
line 1045: C On first year and first iteration calculate base year system load
line 1046:      IF(K1.EQ.FIRSYR .AND. CURITR.EQ.1 ) THEN
line 1047:          flag=.TRUE.
line 1048: C Check if the base year demand has been supplied in the structure file
line 1049:          DO I=1,Neu
line 1050:              IF (BaseYrLd(I,RNB).NE.0.0) flag=.FALSE.
line 1051:          ENDDO

```

```

line 1052:    IF(FLAG) THEN ! if on input all base year loads=0.0 use first year data
line 1053:    DO I=1,Neu
line 1054:        BaseYrLd(I,RNB)=LoadForec(I,1)
line 1055:    ENDDO
line 1056:    ENDIF
line 1057:    load=0.0
line 1058:    DO I=1,Neu
line 1059:        load=load+BaseYrLd(I,RNB)
line 1060:    ENDDO
line 1061:    BaseYrSysLd(RNB)=load
line 1062:    ENDIF
line 1063: C Apply DELTA approach to the load forecast
line 1064:    CALL DSMDLT
line 1065: C***** Termination of the Program/Subprogram *****
line 1066:    RETURN
line 1067: 999 WRITE(IMG,*)<)) Message from subroutine DSMFOR'
line 1068:    WRITE(IMG,*)<)) Access to load forecast for sector ',SLNAM(I)
line 1069:    &,' is undefined'
line 1070:    WRITE(IMG,*)<)) Execution of the program terminated'
line 1071:    WHOOPS=.TRUE.
line 1072:    RETURN
line 1073:    END
line 1074:    SUBROUTINE DSMHLM
line 1075: C***** Description of the Program/Subprogram *****
line 1076: C This subroutine runs the HELM algorithm subroutines
line 1077: C-----
line 1078: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 1079: C*****
line 1080:    IMPLICIT NONE
line 1081: C***** Typing, Declaring and Initializing Constant Parameters ****
line 1082:    INTEGER*2 ZERO,ONE
line 1083:    PARAMETER(ZERO=0,ONE=1)
line 1084:    INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 1085:    INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 1086:    INCLUDE 'EMM$PARM' !<< EMM GLOBAL PARAMETERS
line 1087: C***** Typing and Declaring Variables *****
line 1088:    INTEGER*2 I,J,K,L,M,N ! multi-purpose pointers
line 1089:    REAL*4 td ! t&dloss expansion factor
line 1090:    REAL*4 SystLo !system load
line 1091:    REAL*4 SYLOAD1(MAXHOUR) !matrix with system load sorted in descending ord.
line 1092:    INTEGER*2 Hindex(MAXHOUR) ! indexes of hours
line 1093: C***** COMMON and EQUIVALENCE *****
line 1094:    INCLUDE 'NCNTRL' !<< global NEMS variables
line 1095:    INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 1096:    INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 1097:    INCLUDE 'DSMHLM' !<< HELM algorithm variables
line 1098:    INCLUDE 'DSMNEMSC' !<< results to be passed to the rest of NEMS
line 1099:    INCLUDE 'DSMTFECP' !<< communication with ECP
line 1100:    INCLUDE 'DSMTOefd' !<< communication with EFD
line 1101:    INCLUDE 'DSMTFEFP' !<< communication with EFP
line 1102:    INCLUDE 'DSMNERCR' !<< NERC region data
line 1103:    INCLUDE 'DSMCALDR' !<< calendar data
line 1104: C***** Initializing Variables *****
line 1105:    IF (K1.EQ.CURIYR) THEN
line 1106:        DO I=1,MAXHOUR
line 1107:            DO K=1,MAXSEC
line 1108:                SectorLoad(I,K)=0.0
line 1109:            ENDDO
line 1110:        ENDDO
line 1111:        DO K=1,MAXSEC
line 1112:            TotSecLoad(RNB,K)=0.0
line 1113:        ENDDO

```

```

line 1114:   ENDIF
line 1115:   DO I=1,MAXHOUR
line 1116:     SYLOAD(I)=0.0
line 1117:   ENDDO
line 1118: C***** Body of the Program/Subprogram *****
line 1119: C*** Calculate base system load
line 1120:   IF (SystemLoad.GT.0.0) THEN
line 1121:     CALL DSMNWS(NERCIsrN(RNB),ZERO,ZERO)
line 1122:   ELSE
line 1123:     WRITE(IMG,*)'<')) Warning from subroutine DSMHLM'
line 1124:     WRITE(IMG,*)' System load for region: ',NERCnam(RNB),' year:',
line 1125:     &      K1,' <= 0.0'
line 1126:   ENDIF
line 1127: C*** Now modify base system load according to changes in system structure
line 1128:   K=0 ! pointer on the list of end-uses
line 1129:   SystLo=0.0
line 1130:   DO I=1,NumSec ! do for each sector
line 1131:     DO J=1,NEUSES(I) ! do for each end-use in a sector
line 1132:       K=K+1
line 1133:       CALL DSMNWS(EUrecNUM(K,RNB),K,I)
line 1134:     ENDDO
line 1135:     SystLo=SystLo+TotSecLoad(RNB,I)
line 1136:   ENDDO
line 1137: C Expand load by t&d loss factor
line 1138: C and find system peak, total system load,system load factor
line 1139:   td=1.0+NERCtdloss(RNB)
line 1140:   SystLo=SystLo*td
line 1141:   DO I=1,nhour
line 1142:     SYLOAD(I)=SYLOAD(I)*td
line 1143:     SYLOAD1(I)=SYLOAD(I)
line 1144:     Hindex(I)=I
line 1145:   ENDDO
line 1146:   CALL DSMQSR(SYLOAD1,Hindex,ONE,nhour)
line 1147:   SystemPeak(K1)=SYLOAD1(1)
line 1148:   SystemPeakHour(K1)=Hindex(1)
line 1149:   TotSystemLoad(K1)=SystLo
line 1150:   SystemLoadFactor(K1)=SystLo/nhouryr/SystemPeak(K1)
line 1151:   J=0
line 1152:   DO I=1,nhour
line 1153:     DO K=1,HourlyWeights(Hindex(I))
line 1154:       J=J+1
line 1155:       IF(J.GT.NpeakH) GOTO 300
line 1156:       SysPeakHour(J)=Hindex(I)
line 1157:     ENDDO
line 1158:   ENDDO
line 1159: 300 CONTINUE
line 1160: C***** Termination of the Program/Subprogram *****
line 1161:   RETURN
line 1162:   END
line 1163:   SUBROUTINE DSMLCP(SWITCH)
line 1164: C***** Description of the Program/Subprogram *****
line 1165: C*** This subroutine develops LDC's required by ECP module
line 1166: C-----
line 1167: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 1168: C*****
line 1169:   IMPLICIT NONE
line 1170: C***** Typing, Declaring and Initializing Constant Parameters ***
line 1171:   INTEGER*2 ONE
line 1172:   PARAMETER(ONE=1)
line 1173:   INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 1174:   INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 1175:   INCLUDE 'EMM$PARM' !<< EMM GLOBAL PARAMETERS

```

```

line 1176: C***** Typing and Declaring Variables *****
line 1177: REAL*4 SYLOAD1(MAXHOUR) !copy of SYLOAD for one segment of hour # in year
line 1178: REAL*4 SYLOAD2(MAXHOUR) !same as SYLOAD1 but different order
line 1179: REAL*4 area ! area of the block
line 1180: INTEGER*2 i,j,l,h,h1,h2,w ! temporary variables
line 1181: INTEGER*4 SWITCH ! switch: SYSTEM LOAD or DSM program (is being processed)
line 1182: REAL*4 dh ! division point of an hour
line 1183: REAL*4 pl ! current peak load
line 1184: INTEGER*2 sgn ! segment number
line 1185: INTEGER*2 blk ! number of a block in the entire LDC
line 1186: INTEGER*2 bls ! number of a block in the segment
line 1187: INTEGER*2 hn ! current hour number
line 1188: C***** COMMON and EQUIVALENCE *****
line 1189: INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 1190: INCLUDE 'DSMCALDR' !<< calendar data
line 1191: INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 1192: INCLUDE 'DSMNEMSC' !<< results to be passed to the rest of NEMS
line 1193: INCLUDE 'DSMTFECF' !<< communication with ECP
line 1194: INCLUDE 'DSMNERCR' !<< NERC region data
line 1195: C***** Initializing Variables *****
line 1196: C***** Body of the Program/Subprogram *****
line 1197: C For every segment of load of ECP LDC develop load duration curves
line 1198: C Create a vector of hourly loads for all calendar hours
line 1199: DO hn=1,nhour
line 1200: SYLOAD1(hn)=SYLOAD(hn)
line 1201: ENDDO
line 1202: C Rewrite SYLOAD1 to SYLOAD2 in segment order
line 1203: DO i=1,nhour
line 1204: SYLOAD2(i)=SYLOAD1(HourNumber(i))
line 1205: ENDDO
line 1206: IF(SWITCH.EQ.1) THEN
line 1207: C Sort loads in each segment in descending order
line 1208: DO i=1,ECPnumSg
line 1209: CALL DSMQSR(SYLOAD2,HourNumber,ECPsgFh(i),ECPsgLh(i))
line 1210: ENDDO
line 1211: ENDIF
line 1212: C SYLOAD2 now contains calendar hour loads sorted by segment(from last to first)
line 1213: C then in each segment loads are sorted in descending order
line 1214: C HourNumber contains original positions of calendar hours
line 1215: C Determine heights of the ECP LDC blocks
line 1216: blk=1 ! current block number within whole LDC
line 1217: bls=1 ! current block number within a segment
line 1218: area=0.0 ! current area under the curve from the end of the previous block
line 1219: C Now go over all hours in the year and calculate blocks widths and heights
line 1220: DO sgn=ECPnumSg,1,-1
line 1221: l=0 ! x coordinate expressed in real hours
line 1222: h1=ECPsgFh(sgn) ! beginning of current block in calendar hours
line 1223: h2=ECPsgLh(sgn) ! end of current block in calendar hours
line 1224: pl=SYLOAD2(h1) !current local peak load
line 1225: DO h=h1,h2 ! x coordinate expressed in calendar hours
line 1226: DO w=1,HourlyWeights(HourNumber(h))
line 1227: l=l+1 ! x coordinate in real hours
line 1228: area=area+SYLOAD2(h)
line 1229: IF(blS.NE.ECPsgDnB(sgn)) THEN !if this is not last block in segment
line 1230: IF(l.GT.ECPblockx(sgn,bls)) THEN !if x coordinate beyond the block
line 1231: dh=l-ECPblockx(sgn,bls) ! calculate an x access
line 1232: IF (ECPsgDblyp(sgn,bls).EQ.'p').OR.
line 1233: & ECPsgDblyp(sgn,bls).EQ.'P') THEN !if a 'peak' type block
line 1234: BlockHeight(blk)=pl
line 1235: ELSE
line 1236: BlockHeight(blk)=(area-dh*SYLOAD2(h))/ECPblWidth(blk)
line 1237: ENDIF

```



```

line 1238:      area=dh*SYLOAD2(h)
line 1239:      bls=bls+1
line 1240:      blk=blk+1
line 1241:      pl=SYLOAD2(h)
line 1242:      ENDIF
line 1243:      ENDIF
line 1244:      ENDDO
line 1245:      ENDDO
line 1246:      BlockHeight(blk)=area/ECPblWidth(blk) ! for last block in a segment
line 1247:      blk=blk+1
line 1248:      bls=1
line 1249:      area=0.0
line 1250:      ENDDO
line 1251:      ECPnumBl=blk-1
line 1252:      DO i=1,ECPnumBl
line 1253:      BlockNum(i)=i
line 1254:      ENDDO
line 1255:      IF(SWITCH.EQ.1) THEN
line 1256: C Sort LDC blocks in their height order
line 1257:      CALL DSMQSR(BlockHeight,BlockNum,ONE,blk)
line 1258: C Write into the communication common block arrays:
line 1259: C ECPLDCBH(year,region,blocknumber) heights,
line 1260: C ECPLDCBW(year,region,blocknumber) widths,
line 1261: C ECPLDCBS(year,region,blocknumber) segment identification, for all blocks.
line 1262:      DO i=1,ECPnumBl
line 1263:      ECPLDCBH(K1,RNB,i)=BlockHeight(i)
line 1264:      ECPLDCBW(K1,RNB,i)=ECPblWidth(BlockNum(i))/SumSegWidth !to fr. of year
line 1265:      ECPLDCBS(K1,RNB,i)=ECPblSeg(BlockNum(i))
line 1266:      ENDDO
line 1267:      ENDIF
line 1268: C***** Termination of the Program/Subprogram *****
line 1269:      RETURN
line 1270:      END
line 1271:      REAL*4 FUNCTION DSMNVAL(CENSUSvalues,NERCreg,sector)
line 1272: C***** Description of the Program/Subprogram *****
line 1273: C This routine develops a NERC region value based on the values for the CENSUS
line 1274: C region. It uses different mapping matrices for different demand sectors.
line 1275: C-----
line 1276: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 1277: C*****
line 1278:      IMPLICIT NONE
line 1279: C***** Typing, Declaring and Initializing Constant Parameters ****
line 1280:      INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 1281:      INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 1282:      INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 1283:      INCLUDE 'DSMNRCR' !<< NERC region data
line 1284:      INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 1285: C***** Typing and Declaring Variables *****
line 1286:      REAL*4 CENSUSvalues(MAXCRG)  ! CENSUS division values
line 1287:      INTEGER*2 NERCreg  ! current NERC region
line 1288:      INTEGER*2 sector  ! current sector
line 1289:      INTEGER*2 k  ! universal counter
line 1290: C***** COMMON and EQUIVALENCE *****
line 1291: C***** Initializing Variables *****
line 1292: C***** Body of the Program/Subprogram *****
line 1293:      DSMNVAL=0.0
line 1294:      DO k=1,nCENSUSreg
line 1295:      DSMNVAL=DSMNVAL+CENSUSvalues(k)*MappCtoN(NERCreg,k,sector)
line 1296:      ENDDO
line 1297: C***** Termination of the Program/Subprogram *****
line 1298:      RETURN
line 1299:      END

```

```

line 1300:  SUBROUTINE DSMNWS(NUMREC,LFpointer,SECTOR)
line 1301: C***** Description of the Program/Subprogram *****
line 1302: C This subroutine adds up current item's load to the system load
line 1303: C-----
line 1304: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 1305: C*****
line 1306:  IMPLICIT NONE
line 1307: C***** Typing, Declaring and Initializing Constant Parameters ****
line 1308:  INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 1309: C***** Typing and Declaring Variables *****
line 1310:  INTEGER*2 K ! month pointer
line 1311:  INTEGER*2 L ! day pointer
line 1312:  INTEGER*2 M ! hour pointer
line 1313:  INTEGER*2 SECTOR ! sector number or 0 if not to process sectoral loads
line 1314:  INTEGER*2 LFpointer ! pointer to LoadForec array or 0 if SystemLoad
line 1315:  INTEGER*2 NUMREC ! number of record and DAF-LSR-DB with current lsr
line 1316:  REAL*4 DistLo(MAXHOUR) ! distribution of annual load over hours
line 1317:  REAL*8 load1 ! temporary variables for annual load forecast orig. appr.
line 1318:  REAL*8 load2 ! temporary variables for annual load forecast delta appr.
line 1319:  CHARACTER*8 LSRname ! Current lsr name
line 1320: C***** COMMON and EQUIVALENCE *****
line 1321:  INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 1322:  INCLUDE 'NCNTRL' !<< global NEMS variables
line 1323:  INCLUDE 'DSMCALDR' !<< calendar data
line 1324:  INCLUDE 'DSMHLM' !<< HELM algorithm variables
line 1325:  INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 1326:  INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 1327:  INCLUDE 'DSMTFEFP' !<< communication with EFP
line 1328: C***** Initializing Variables *****
line 1329: C***** Body of the Program/Subprogram *****
line 1330: C Read a record from the Direct Access File
line 1331:  READ(IODB,REC=NUMREC)LSRname,
line 1332:  & (DistLo(M),M=1,nhour)
line 1333:  IF(LFpointer.NE.0) THEN
line 1334:    load1=LoadForec(LFpointer,1)
line 1335:    load2=LoadForec(LFpointer,2)
line 1336:  ELSE
line 1337:    load2=SystemLoad      !if the item is a system load
line 1338:  ENDIF
line 1339:  IF(load2.NE.0.0) THEN
line 1340:    DO M=1,nhour
line 1341:      SYLOAD(M)=SYLOAD(M)+DistLo(M)*load2
line 1342:    ENDDO
line 1343:  ENDIF
line 1344:  IF(SECTOR.NE.0 .AND. K1.EQ.CURIYR) THEN
line 1345:    DO M=1,nhour
line 1346:      SectorLoad(M,SECTOR)=SectorLoad(M,SECTOR)
line 1347:      & +DistLo(M)*load1
line 1348:    ENDDO
line 1349:    TotSecLoad(RNB,SECTOR)=TotSecLoad(RNB,SECTOR)+load1
line 1350:  ENDIF
line 1351: C***** Termination of the Program/Subprogram *****
line 1352:  RETURN
line 1353:  END
line 1354:  SUBROUTINE DSMPCIM(WHOOPS)
line 1355: C***** Description of the Program/Subprogram *****
line 1356: C This subroutine for both Residential and Commercial sectors calculates:
line 1357: C 1) actualized costs of DSM programs
line 1358: C 2) total annual load impact of DSM programs
line 1359: C 3) invokes TRC tests
line 1360: C-----
line 1361: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353

```

```

line 1362: C*****
line 1363: IMPLICIT NONE
line 1364: C***** Typing, Declaring and Initializing Constant Parameters ****
line 1365: INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 1366: INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 1367: INCLUDE 'EMM$PARM' !<< EMM GLOBAL PARAMETERS
line 1368: INTEGER*2 NUMBYR
line 1369: INTEGER*2 NYREST1,NYREST2
line 1370: PARAMETER(NUMBYR=MNUMYR+ECP$XPH)
line 1371: PARAMETER(NYREST1=NYRESTR+ECP$XPH-1)
line 1372: PARAMETER(NYREST2=NYRESTC+ECP$XPH-1)
line 1373: C***** Typing and Declaring Variables *****
line 1374: INTEGER*4 IDENTIF
line 1375: REAL*4 uecc(MAXCTECH) ! unitary energy cons. for commercial techn.
line 1376: REAL*4 UEC(NYRESTR,MAXDMT) !UEC's PER SINGLE appliance
line 1377: REAL*4 MKS(NYREST1,MAXRDT,MAXDMT) !MARKET SIZE BY DECISION TYPE (# APPL.)
line 1378: REAL*4 CNS(NYRESTR,MAXDMT) !SECTORAL CONSUMPTION OF ELECTRICITY
line 1379: REAL*4 CNSC(NYRESTC,MAXNTPO) ! SECTORAL CONSUMP. OF ELECTRICITY IN COMM.
line 1380: INTEGER*2 nopt ! number of options in a program
line 1381: INTEGER*2 opti ! option index on the list of options
line 1382: INTEGER*2 ntech ! number of technologies
line 1383: INTEGER*2 buildi ! building type index
line 1384: INTEGER*2 decti ! decision type index
line 1385: INTEGER*2 ti ! technology index
line 1386: INTEGER*2 tdn ! demand module technology number
line 1387: INTEGER*2 I,J,K,L,Y,D,C,YR,YRR,YY
line 1388: INTEGER*2 SYR ! START YEAR OF THE PROGRAM
line 1389: INTEGER*2 SYRix ! start year index
line 1390: INTEGER*2 PYR ! PROGRAM YEAR
line 1391: INTEGER*2 PYRix ! program year index
line 1392: INTEGER*2 fTO ! index of the first TO technology
line 1393: REAL*4 cost(ECP$XPH) ! total actualized cost of the program
line 1394: REAL*4 NumbAppl(NUMBYR,2)! 1,Number of appliances to be shifted due to DSM
line 1395: ! 2,Number of free-riders associated with DSM
line 1396: REAL*4 NumbApplCum ! Cumulative number of appliances Gained to TO tech.
line 1397: REAL*4 ServDem(NUMBYR,2) ! 1,Service Demand to be shifted due to DSM
line 1398: ! 2,Service Demand associated with free-riders
line 1399: REAL*4 ServDemCum ! Service Demand gained to the TO technologies
line 1400: REAL*4 RMS ! resulting market share of the TO technology (fraction)
line 1401: INTEGER*2 ramp ! length of ramp-up period
line 1402: REAL*4 pramp ! fraction of gains due to ramp-up period (years)
line 1403: REAL*4 ppramp ! cumulative fraction of gains
line 1404: REAL*4 df ! discount coefficient
line 1405: REAL*4 RATIO ! foresight adjustment factor for current year and region
line 1406: INTEGER*2 n,nt ! technology pointers
line 1407: INTEGER*2 nta,ntb ! number of technologies affected by current program
line 1408: REAL*4 CNSFR(ECP$XPH) ! Sectoral energy cons. from FORESIGHT, NERC reg.
line 1409: REAL*4 DSMRMS ! FUNCTION WHICH CALCULATES MARKET SHARES
line 1410: EXTERNAL DSMRMS
line 1411: REAL*4 REBATE ! REBATE
line 1412: REAL*4 OneYearEnSav !annual energy savings induced by DSM option per appl.
line 1413: INTEGER*4 RECN ! RECORD NUMBER ON RESTARTR FILE
line 1414: REAL*4 SDEM(NYRESTC,MAXDECT,MAXCTECH) ! Service Demand for comm. technol.
line 1415: REAL*4 teff ! technology efficiency
line 1416: INTEGER*2 tgn ! technology group number
line 1417: LOGICAL*1 PassOrFail ! Passed or failed TRC test?
line 1418: LOGICAL WHOOPS ! error flag
line 1419: INTEGER*2 CYR1 ! year preceding current year
line 1420: REAL*4 MMKSF,MMKST ! minimum market share for FROM and for TO
line 1421: REAL*4 TotNumbAppl ! total number of appliances that may be shifted
line 1422: REAL*4 TotServDem ! total service demand that may be shifted
line 1423: REAL*4 CurrNA ! current number of appliances

```

```

line 1424: REAL*4 CurrSD ! current amount of service demand
line 1425: REAL*4 CurrFR ! current free-riders
line 1426: REAL*4 Perc ! fraction of the market covered by already running DSM prog.
line 1427: REAL*4 CENSUSvalues(MAXCRG) ! temporary vector with CENSUS division values
line 1428: EXTERNAL DSMNVAL
line 1429: C***** COMMON and EQUIVALENCE *****
line 1430: INCLUDE 'NCNTRL' !<< global MEMS variables
line 1431: INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 1432: INCLUDE 'DSMPROG' !<< DSM program data
line 1433: INCLUDE 'QBLK' !<< supplies some constants needed in MXQBLK
line 1434: INCLUDE 'MXQBLK' !<< foresight data
line 1435: INCLUDE 'DSMNERCR' !<< NERC region data
line 1436: INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 1437: INCLUDE 'DSMOPTDB' !<< DSM options data base
line 1438: INCLUDE 'DSMNEMSC' !<< results of LDSM to be passed to the rest of NEMS
line 1439: INCLUDE 'DSMTFECF' !<< communication with ECP
line 1440: INCLUDE 'COMPARM' !<< defines PARAMETERS: CMnumBldg,CMnumServ
line 1441: INCLUDE 'COMVARS' !<< defines Parameter: CMnumRule
line 1442: INCLUDE 'DSMCMTDB' !<< COMMERCIAL techn. database
line 1443: INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 1444: INCLUDE 'DSMCALDR' !<< calendar data
line 1445: C***** Initializing Variables *****
line 1446: C***** Body of the Program/Subprogram *****
line 1447: C
line 1448: C FOR RESIDENTIAL SECTOR
line 1449: C
line 1450: C
line 1451: C MAP FORESIGHT DATA INTO NERC REGION
line 1452: DO YR=1,ECP$XPH
line 1453: Y=YR+CURIYR-1
line 1454: DO C=1,nCENSUSreg
line 1455: CENSUSvalues(k)=XQELRS(C,Y)
line 1456: ENDDO
line 1457: CNSFR(YR)=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 1458: & *UNCONFA
line 1459: ENDDO
line 1460: C
line 1461: C CALCULATE AFTER-DSM PERCENTAGE MARKET SHARE OF THE "TO" TECHNOLOGIES
line 1462: C AS A PERCENTAGE OF THE "TO" + THE "FROM" TECHNOLOGY
line 1463: RMS=DSMRMS(PAYBACK,1)
line 1464: C
line 1465: DO I=1,NRPROG ! FOR EACH RESIDENTIAL PROGRAM
line 1466: DO J=1,ECP$XPH
line 1467: cost(J)=0.0
line 1468: ENDDO
line 1469: nta=0
line 1470: ntb=0
line 1471: nopt=RPROGDEFN(RNB,I)
line 1472: DO J=1,nopt ! FOR EACH OPTION IN THE PROGRAM
line 1473: opti=RPROGDEF(RNB,I,J,1)
line 1474: buildi=DSMROptionBuildT(opti)
line 1475: decti=DSMROptionDecType(opti)
line 1476: pramp=1.0/DSMROptionRamp(opti)
line 1477: nt=0
line 1478: n=0
line 1479: C READ APPROPRIATE RECORDS FROM THE RESTART FILE FOR THE 'FROM' TECHNOLOGIES
line 1480: ntech=DSMROptionFromTnum(opti)
line 1481: DO K=1,ntech ! FOR EACH "FROM" DSM-DB TECHNOLOGY OF THE OPTION
line 1482: ti=DSMROptionFromTech(opti,K)
line 1483: tdn=RtechDMtn(ti)
line 1484: DO L=1,tdn ! FOR EACH DEMAND MODULE TECHNOLOGY OF THE DSM-DB TECH.
line 1485: RECN=RtechRrc(ti,L,RNB,buildi)

```

```

line 1486:      tgn=RtechDMG(ti,L)
line 1487:      nt=nt+1
line 1488:      nta=nta+1
line 1489:      IF(RECN.NE.0) THEN
line 1490:          READ(IORR,REC=RECN)IDENTIF,((MKS(Y,D,nt),D=1,RdecTYPn),
line 1491:      &          UEC(Y,nt),CNS(Y,nt),Y=1,NYRESTR)
line 1492:      ELSE
line 1493:          nta=nta-nt
line 1494:          RPROGDEF(RNB,I,J,2)=1
line 1495:          GOTO 1231
line 1496:      ENDIF
line 1497:      LSRindex(nta,I)=RtechLSR(ti)
line 1498: C UPDATE THE INFORMATION FROM RESTART FILE ACCORDING TO FORESIGHT FILE DATA
line 1499:      DO Y=CURIYR+1,ECPLastYearIndex
line 1500:          YR=Y-CURIYR+1
line 1501:          YRR=Y-1
line 1502:          IF(YRR.LE.NYRESTR) THEN ! IF RESTART FILE DATA AVAILABLE
line 1503: C CALCULATE ADJUSTMENT RATIO ACCORDING TO THE FORESIGHT DATA
line 1504:          IF(CNS(YRR,nt).EQ.0.0) GOTO 999
line 1505:          RATIO=CNSFR(YR)/CNSFR(1)*ResTGdem(tgn)/CNS(YRR,nt)
line 1506: C ADJUST MKS AND UEC ACCORDING TO FORESIGHT DATA
line 1507:          DO D=1,RdecTYPn
line 1508:              MKS(YRR,D,nt)=MKS(YRR,D,nt)*RATIO
line 1509:          ENDDO
line 1510:      ELSE
line 1511:          RATIO=CNSFR(YR)/CNSFR(NYRESTR-CURIYR+2)
line 1512:          DO D=1,RdecTYPn
line 1513:              MKS(YRR,D,nt)=MKS(NYRESTR,D,nt)*RATIO
line 1514:          ENDDO
line 1515:      ENDIF
line 1516:      ENDDO
line 1517:      ENDDO
line 1518:      ENDDO
line 1519:      fTO=nt+1
line 1520: C READ APPROPRIATE RECORDS FROM THE RESTART FILE FOR THE 'TO' TECHNOLOGIES
line 1521:      ntech=DSMROptionToTnum(opti)
line 1522:      DO K=1,ntech ! FOR EACH "TO" DSM-DB TECHNOLOGY OF THE OPTION
line 1523:          ti=DSMROptionToTech(opti,K)
line 1524:          tdn=RtechDMtn(ti)
line 1525:          DO L=1,tdn ! FOR EACH DEMAND MODULE TECHNOLOGY OF THE DSM-DB TECH.
line 1526:              RECN=RtechRrc(ti,L,RNB,buildi)
line 1527:              tgn=RtechDMG(ti,L)
line 1528:              nt=nt+1
line 1529:              nta=nta+1
line 1530:              IF (RECN.NE.0) THEN
line 1531:                  READ(IORR,REC=RECN)IDENTIF,((MKS(Y,D,nt),D=1,RdecTYPn),
line 1532:      &                  UEC(Y,nt),CNS(Y,nt),Y=1,NYRESTR)
line 1533:              ELSE
line 1534:                  RPROGDEF(RNB,I,J,2)=1
line 1535:                  nta=nta-nt
line 1536:                  GOTO 1231
line 1537:              ENDIF
line 1538:              LSRindex(nta,I)=RtechLSR(ti)
line 1539: C UPDATE THE INFORMATION FROM RESTART FILE ACCORDING TO FORESIGHT FILE DATA
line 1540:              DO Y=CURIYR+1,ECPLastYearIndex
line 1541:                  YR=Y-CURIYR+1
line 1542:                  YRR=Y-1
line 1543:                  IF(YRR.LE.NYRESTR) THEN ! IF RESTART FILE DATA AVAILABLE
line 1544: C CALCULATE ADJUSTMENT RATIO ACCORDING TO THE FORESIGHT DATA
line 1545:                  IF(CNS(YRR,nt).EQ.0.0) GOTO 999
line 1546:                  RATIO=CNSFR(YR)/CNSFR(1)*ResTGdem(tgn)/CNS(YRR,nt)
line 1547: C ADJUST MKS AND UEC ACCORDING TO FORESIGHT DATA

```

```

line 1548:      DO D=1,RdecTYPn
line 1549:      MKS(YRR,D,nt)=MKS(YRR,D,nt)*RATIO
line 1550:      ENDDO
line 1551:      ELSE
line 1552:      RATIO=CNSFR(YR)/CNSFR(NYRESTR-CURIYR+2)
line 1553:      DO D=1,RdecTYPn
line 1554:      MKS(YRR,D,nt)=MKS(NYRESTR,D,nt)*RATIO
line 1555:      ENDDO
line 1556:      ENDIF
line 1557:      ENDDO
line 1558:      ENDDO
line 1559:      ENDDO
line 1560: C PERFORM A TRC TEST AND CALCULATE REBATE FOR THE OPTION
line 1561:      CALL DSMTRCR(opti,UEC,PassOrFail,REBATE)
line 1562:      IF(.NOT.PassOrFail) THEN
line 1563:      RPROGDEF(RNB,I,J,2)=REBATE
line 1564:      nta=nta-nt
line 1565:      GOTO 1231
line 1566:      ENDIF
line 1567: C
line 1568: C CALCULATE NUMBER OF appliances TO BE SHIFTED TO THE 'TO' TECHNOLOGIES FROM
line 1569: C EACH OF THE 'FROM TECHNOLOGIES' AS IT COMES OUT OF THE 'ACCEPTANCE CURVE'
line 1570: C AND CONSIDERING PROGRAMS ALREADY STARTED IN THE PAST
line 1571: C
line 1572:      DO Y=CURIYR+1,ECPLastYearIndex
line 1573: C CALCULATE PERCENTAGE OF THE MARKET COVERED BY THE PROGRAMS ALREADY STARTED
line 1574:      YRR=Y-1
line 1575:      Perc=0.0
line 1576:      CYR1=CURIYR-1
line 1577:      DO YY=FIRSYR+1,CYR1
line 1578:      ppramp=pramp*(Y-YY) ! option starts one year after decision
line 1579:      IF(ppramp.GT.1.0) ppramp=1.0
line 1580:      Perc=Perc+ppramp*RopChoice(YY,RNB,opti)
line 1581:      ENDDO
line 1582:      IF(Perc.GT.1.0) GOTO 996
line 1583: C Determine current market size for an option
line 1584: C find minimum size of FROM technologies
line 1585:      MMKSF=1.0E+30
line 1586:      DO K=1,fTO-1
line 1587:      IF(MKS(YRR,decti,K).LT.MMKSF) MMKSF=MKS(YRR,decti,K)
line 1588:      ENDDO
line 1589: C find minimum size of TO technologies
line 1590:      MMKST=1.0E+30
line 1591:      DO K=fTO,nt
line 1592:      IF(MKS(YRR,decti,K).LT.MMKST) MMKST=MKS(YRR,decti,K)
line 1593:      ENDDO
line 1594: C number of appliances that may be shifted by any old and new DSM PROGRAM
line 1595:      TotNumbAppl=(MMKSF+MMKST)*RMS-MMKST
line 1596:      IF (TotNumbAppl.LT.0.0) THEN
line 1597:      TotNumbAppl=0.0
line 1598:      ENDIF
line 1599: C Maximum number of appliances subject to a shift decision at a given year
line 1600:      NumbAppl(Y,1)=TotNumbAppl*(1.0-Perc)
line 1601:      NumbAppl(Y,2)=MMKST*(1.0-Perc) ! free-riders
line 1602:      ENDDO
line 1603:      DO SYR=CURIYR+1,ECPLastYearIndex ! FOR EVERY POSSIBLE PROGRAM START YR
line 1604:      SYRix=SYR-CURIYR
line 1605:      NumbApplCum=0
line 1606:      df=1.0
line 1607:      DO PYR=SYR,ECPLastYearIndex ! FOR EVERY PROGRAM YEAR
line 1608:      PYRix=PYR-CURIYR
line 1609:      YRR=PYR-1

```

```

line 1610:      IF(YRR.GT.NYRESTR) YRR=NYRESTR
line 1611:      df=df/(1.0+DISCFA)
line 1612:      OneYearEnSav=0.0
line 1613:      IF (PYRix-SYRix+1.LT.DSMROptionRamp(opti)) THEN
line 1614:          CurrNA=NumbAppl(PYR,1)*pramp*(PYRix-SYRix+1)
line 1615:          CurrFR=NumbAppl(PYR,2)*pramp*(PYRix-SYRix+1)
line 1616:      ELSE
line 1617:          CurrNA=NumbAppl(PYR,1)
line 1618:          CurrFR=NumbAppl(PYR,2)
line 1619:      ENDIF
line 1620:      NumbApplCum=NumbApplCum+CurrNA
line 1621:      DO L=1,fTO-1
line 1622:          OneYearEnSav=OneYearEnSav+UEC(YRR,L)
line 1623:          DeltaEnergy(L+ntb,SYRix,PYRix,I)=NumbApplCum*UEC(YRR,L)
line 1624:      ENDDO
line 1625:      DO L=fTO,nt
line 1626:          OneYearEnSav=OneYearEnSav-UEC(YRR,L)
line 1627:          DeltaEnergy(L+ntb,SYRix,PYRix,I)=-NumbApplCum*UEC(YRR,L)
line 1628:      ENDDO
line 1629:      cost(SYRix)=cost(SYRix)+(CurrNA+CurrFR)*(REBATE+
line 1630: &      OneYearEnSav*DSMROptionCost(opti))*df
line 1631:      ENDDO
line 1632:      DSMPRCST(SYRix,RNB,I)=cost(SYRix)
line 1633:      ENDDO
line 1634:      ntb=ntb+nt
line 1635: 1231  CONTINUE
line 1636:      ENDDO
line 1637:      IF(nta.NE.ntb) GOTO 997
line 1638:      NTECHAFFP(I)=nta
line 1639:      ENDDO
line 1640: C+++++ FOR COMMERCIAL PROGRAMS ++++++
line 1641: C MAP FORESIGHT DATA INTO NERC REGION
line 1642:      DO YR=1,ECP$XPH
line 1643:          Y=YR+CURIYR-1
line 1644:          DO C=1,nCENSUSreg
line 1645:              CENSUSvalues(k)=XQELCM(C,Y)
line 1646:          ENDDO
line 1647:          CNSFR(YR)=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 1648:      &      *UNCONFA
line 1649:      ENDDO
line 1650: C CALCULATE AFTER-DSM PERCENTAGE MARKET SHARE OF THE "TO" TECHNOLOGIES
line 1651: C AS A PERCENTAGE OF THE "TO" + THE "FROM" TECHNOLOGY
line 1652:      RMS=DSMRMS(PAYBACK,2)
line 1653:      DO I=1,NCPROG ! FOR EACH COMMERCIAL PROGRAM
line 1654:          DO J=1,ECP$XPH
line 1655:              cost(J)=0.0
line 1656:          ENDDO
line 1657:          nta=0
line 1658:          ntb=0
line 1659:          nopt=CPROGDEFN(RNB,I)
line 1660:          DO J=1,nopt ! FOR EACH OPTION IN THE PROGRAM
line 1661:              opti=CPROGDEF(RNB,I,J,1)
line 1662:              buildi=DSMCOptionBuildT(opti)
line 1663:              decTi=DSMCOptionDecType(opti)
line 1664:              pramp=1.0/DSMCOptionRamp(opti)
line 1665:              nt=0
line 1666:              n=0
line 1667: C READ APPROPRIATE RECORDS FROM THE RESTART FILE FOR THE 'FROM' TECHNOLOGIES
line 1668:          ntech=DSMCOptionFromTnum(opti)
line 1669:          DO L=1,ntech ! FOR EACH "FROM" DSM-DB TECHNOLOGY OF THE OPTION
line 1670:              ti=DSMCOptionFromTech(opti,L)
line 1671:              RECN=CtechRrc(ti,RNB,buildi)

```

```

line 1672:      nt=nt+1
line 1673:      nta=nta+1
line 1674:      IF(RECN.NE.0) THEN
line 1675:          READ(IOCR,REC=RECN)IDENTIF,((SDEM(Y,D,nt),D=1,MAXDECT),
line 1676:      &          CNSC(Y,nt),Y=1,NYRESTC)
line 1677:      ELSE
line 1678:          CPROGDEF(RNB,I,J,2)=7
line 1679:          nta=nta-nt
line 1680:          GOTO 1232
line 1681:      ENDIF
line 1682:      LSRindex(nta,I+NRPROG)=CtechLSR(ti,buildi)
line 1683: C UPDATE THE INFORMATION FROM RESTART FILE ACCORDING TO FORESIGHT FILE DATA
line 1684:      DO Y=CURIYR+1,ECPLastYearIndex
line 1685:          YR=Y-CURIYR+1
line 1686:          IF(Y.LE.NYRESTC) THEN ! IF RESTART FILE DATA AVAILABLE
line 1687: C CALCULATE ADJUSTMENT RATIO ACCORDING TO THE FORESIGHT DATA
line 1688:          IF(CNSC(Y,nt).EQ.0.0) GOTO 999
line 1689:          RATIO=1 ! ASSUMING NO CHANGE IN DEMAND FORECAST
line 1690: C          RATIO=CNSFR(YR)/CNSC(Y,nt)
line 1691: C ADJUST SDEM ACCORDING TO FORESIGHT DATA
line 1692:          DO D=1,CdecTYPn
line 1693:              SDEM(Y,D,nt)=SDEM(Y,D,nt)*RATIO
line 1694:          ENDDO
line 1695:      ELSE
line 1696:          RATIO=CNSFR(YR)/CNSFR(NYRESTC-CURIYR+1)
line 1697:          DO D=1,CdecTYPn
line 1698:              SDEM(Y,D,nt)=SDEM(NYRESTC,D,nt)*RATIO
line 1699:          ENDDO
line 1700:      ENDIF
line 1701:      ENDDO
line 1702:      ENDDO
line 1703:      fTO=nt+1
line 1704: C READ APPROPRIATE RECORDS FROM THE RESTART FILE FOR THE 'TO' TECHNOLOGIES
line 1705:      ntech=DSMCOptionToTnum(opti)
line 1706:      DO L=1,ntech ! FOR EACH "TO" DSM-DB TECHNOLOGY OF THE OPTION
line 1707:          ti=DSMCOptionToTech(opti,L)
line 1708:          nt=nt+1
line 1709:          nta=nta+1
line 1710:          LSRindex(nta,I+NRPROG)=CtechLSR(ti,buildi)
line 1711:          RECN=CtechRrc(ti,RNB,buildi)
line 1712:          IF(RECN.NE.0) THEN
line 1713:              READ(IOCR,REC=RECN)IDENTIF,((SDEM(Y,D,nt),D=1,MAXDECT),
line 1714:      &              CNSC(Y,nt),Y=1,NYRESTC)
line 1715:          ELSE
line 1716:              DO Y=1,NYRESTC
line 1717:                  DO D=1,MAXDECT
line 1718:                      SDEM(Y,D,nt)=0.0
line 1719:                  ENDDO
line 1720:                  CNSC(Y,nt)=1.0
line 1721:              ENDDO
line 1722:          ENDIF
line 1723: C UPDATE THE INFORMATION FROM RESTART FILE ACCORDING TO FORESIGHT FILE DATA
line 1724:      DO Y=CURIYR+1,ECPLastYearIndex
line 1725:          YR=Y-CURIYR+1
line 1726:          IF(Y.LE.NYRESTC) THEN
line 1727: C CALCULATE ADJUSTMENT RATIO ACCORDING TO THE FORESIGHT DATA
line 1728:          IF(CNSC(Y,nt).EQ.0.0) GOTO 999
line 1729:          RATIO=1 !ASSUMING NO CHANGE IN DEMAND FORECAST
line 1730: C          RATIO=CNSFR(YR)/CNSC(Y,nt)
line 1731: C ADJUST SDEM ACCORDING TO FORESIGHT DATA
line 1732:          DO D=1,CdecTYPn
line 1733:              SDEM(Y,D,nt)=SDEM(Y,D,nt)*RATIO

```



```

line 1734:      ENDDO
line 1735:      ELSE
line 1736:      RATIO=CNSFR(YR)/CNSFR(NYRESTC-CURIYR+1)
line 1737:      DO D=1,CdecTYPn
line 1738:          SDEM(Y,D,nt)=SDEM(NYRESTC,D,nt)*RATIO
line 1739:      ENDDO
line 1740:      ENDIF
line 1741:      ENDDO
line 1742:      ENDDO
line 1743: C PERFORM A TRC TEST AND CALCULATE REBATE FOR THE OPTION
line 1744:      CALL DSMTRCC(opti,PassOrFail,REBATE,uecc)
line 1745:      IF(.NOT.PassOrFail) THEN
line 1746:          CPROGDEF(RNB,I,J,2)=REBATE
line 1747:          nta=nta-nt
line 1748:          GOTO 1232
line 1749:      ENDIF
line 1750: C
line 1751: C CALCULATE AMOUNT OF SERVICE DEMAND TO BE SHIFTED TO THE 'TO' TECHNOLOGIES FROM
line 1752: C EACH OF THE 'FROM TECHNOLOGIES' AS IT COMES OUT OF THE 'ACCEPTANCE CURVE'
line 1753: C AND CONSIDERING PROGRAMS ALREADY STARTED
line 1754: C
line 1755:      DO Y=CURIYR+1,ECPLastYearIndex
line 1756: C CALCULATE PERCENTAGE OF THE MARKET COVERED BY THE PROGRAMS ALREADY STARTED
line 1757:      Perc=0.0
line 1758:      CYR1=CURIYR-1
line 1759:      DO YY=FIRSYR+1,CYR1
line 1760:          ppramp=pramp*(Y-YY)
line 1761:          IF(ppramp.GT.1.0) ppramp=1.0
line 1762:          Perc=Perc+ppramp*CopChoice(YY,RNB,opti)
line 1763:      ENDDO
line 1764:      IF(Perc.GT.1.0) GOTO 995
line 1765: C Determine current market size for an option
line 1766: C Find minimum size of technologies
line 1767:      MMKSF=1.0E+30
line 1768:      DO K=1,fTO-1
line 1769:          IF(SDEM(Y,decti,K).LT.MMKSF) MMKSF=SDEM(Y,decti,K)
line 1770:      ENDDO
line 1771: C find minimum size of TO technologies
line 1772:      MMKST=1.0E+30
line 1773:      DO K=fTO,nt
line 1774:          IF(SDEM(Y,decti,K).LT.MMKST) MMKST=SDEM(Y,decti,K)
line 1775:      ENDDO
line 1776: C Service demand that may be shifted by any old and new DSM PROGRAM
line 1777:      TotServDem=(MMKSF+MMKST)*RMS-MMKST
line 1778:      IF (TotServDem.LT.0.0) THEN
line 1779:          TotServDem=0.0
line 1780:      ENDIF
line 1781: C Maximum amount of service demand subject to a shift decision at a given year
line 1782:      ServDem(Y,1)=TotServDem*(1.0-Perc)
line 1783:      ServDem(Y,2)=MMKST*(1.0-Perc)
line 1784:      ENDDO
line 1785: C Calculate program impacts and costs
line 1786:      DO SYR=CURIYR+1,ECPLastYearIndex ! FOR EVERY POSSIBLE PROGRAM START YR
line 1787:          SYRix=SYR-CURIYR
line 1788:          ServDemCum=0
line 1789:          df=1.0
line 1790:      DO PYR=SYR,ECPLastYearIndex ! FOR EVERY PROGRAM YEAR
line 1791:          PYRix=PYR-CURIYR
line 1792:          df=df/(1.0+DISCFA)
line 1793:          OneYearEnSav=0.0
line 1794:          IF (PYRix-SYRix+1.LT.DSMCOptionRamp(opti)) THEN
line 1795:              CurrSD=ServDem(PYR,1)*pramp*(PYRix-SYRix+1) !GWhs.d./yr

```

```

line 1796:      CurrFR=ServDem(PYR,2)*pramp*(PYRix-SYRix+1) !GWhs.d./yr
line 1797:      ELSE
line 1798:      CurrSD=ServDem(PYR,1)
line 1799:      CurrFR=ServDem(PYR,2)
line 1800:      ENDDIF
line 1801:      ServDemCum=ServDemCum+CurrSD !GWhs.d./yr
line 1802:      DO L=1,fTO-1
line 1803:      OneYearEnSav=OneYearEnSav+uecc(L) !GWh/yr per GWhs.d/yr
line 1804:      DeltaEnergy(L+ntb,SYRix,PYRix,I+NRPROG)=
line 1805:  &      +ServDemCum*uecc(L) !GWh/yr
line 1806:      ENDDO
line 1807:      DO L=fTO,nt
line 1808:      OneYearEnSav=OneYearEnSav-uecc(L) !GWh/yr per GWhs.d/yr
line 1809:      DeltaEnergy(L+ntb,SYRix,PYRix,I+NRPROG)=
line 1810:  &      -ServDemCum*uecc(L) !GWh/yr
line 1811:      ENDDO
line 1812:      cost(SYRix)=cost(SYRix)+(CurrSD+CurrFR)
line 1813:  &      *(REBATE+OneYearEnSav*DSMCOptionCost(opti))*df
line 1814:      ENDDO
line 1815:      DSMPRCST(SYRix,RNB,NRPROG+I)=cost(SYRix)
line 1816:      ENDDO
line 1817:      ntb=ntb+nt
line 1818: 1232  CONTINUE
line 1819:      ENDDO
line 1820:      IF(nta.NE.ntb) GOTO 997
line 1821:      NTECHAFP(NRPROG+I)=nta
line 1822:      ENDDO
line 1823: C***** Termination of the Program/Subprogram *****
line 1824:      RETURN
line 1825: 999  WRITE(IMG,*)'<')) Message from subroutine DSMPCIM'
line 1826:      WRITE(IMG,*)'<')) Zero sectorial consumption on RESTART file'
line 1827:      WRITE(IMG,*)'<')) Program terminated'
line 1828:      WHOOPS=.TRUE.
line 1829:      RETURN
line 1830: 997  WRITE(IMG,*)'<')) Message from subroutine DSMPCIM'
line 1831:      WRITE(IMG,*)'<')) Program error nta.ne.ntb, region:'nta,ntb,RNB
line 1832:      WRITE(IMG,*)'<')) Program terminated'
line 1833:      WHOOPS=.TRUE.
line 1834:      RETURN
line 1835: 996  WRITE(IMG,*)'<')) Message from subroutine DSMPCIM'
line 1836:      WRITE(IMG,*)'<')) Option implementation over 100% '
line 1837:      &,DSMROptionCode(opti)
line 1838:      WRITE(IMG,*)'<')) Program terminated'
line 1839:      WHOOPS=.TRUE.
line 1840:      RETURN
line 1841: 995  WRITE(IMG,*)'<')) Message from subroutine DSMPCIM'
line 1842:      WRITE(IMG,*)'<')) Option implementation over 100% '
line 1843:      &,DSMCOptionCode(opti)
line 1844:      WRITE(IMG,*)'<')) Program terminated'
line 1845:      WHOOPS=.TRUE.
line 1846:      RETURN
line 1847:      END
line 1848:      SUBROUTINE DSMPRGD
line 1849: C***** Description of the Program/Subprogram *****
line 1850: C This routine defines DSM Programs which are to compete with supply options
line 1851: C within the ECP model
line 1852: C-----
line 1853: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 1854: C*****
line 1855:      IMPLICIT NONE
line 1856: C***** Typing, Declaring and Initializing Constant Parameters ****
line 1857:      INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations

```

```

line 1858:    INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 1859:    INCLUDE 'EMM$PARM' !<< EMM GLOBAL PARAMETERS
line 1860: C***** Typing and Declaring Variables *****
line 1861:    INTEGER*2 i,j,k ! counters
line 1862:    INTEGER*2 IPROG ! program pointer
line 1863:    INTEGER*2 IOPTION ! option pointer
line 1864:    LOGICAL*1 PASSorFAIL ! PASS or FAIL TRC test flag
line 1865: C***** COMMON and EQUIVALENCE *****
line 1866:    INCLUDE 'NCNTRL' !<< global MEMS variables
line 1867:    INCLUDE 'DSMOPTDB' !<< DSM options data base
line 1868:    INCLUDE 'DSMPROG' !<< DSM program data
line 1869:    INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 1870:    INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 1871: C***** Initializing Variables *****
line 1872:    DO IPROG=1,MAXDSMP
line 1873:        DO j=1,MAXOPR
line 1874:            RPROGDEF(RNB,IPROG,j,1)=0
line 1875:            CPROGDEF(RNB,IPROG,j,1)=0
line 1876:            RPROGDEF(RNB,IPROG,j,2)=6
line 1877:            CPROGDEF(RNB,IPROG,j,2)=6
line 1878:        ENDDO
line 1879:    ENDDO
line 1880: C***** Body of the Program/Subprogram *****
line 1881: C FOR RESIDENTIAL PROGRAMS
line 1882:    DO IPROG=1,NRPROG ! do for each residential program
line 1883:        j=0 ! counter of number of options accepted to a program
line 1884:        DO IOPTION=1,DSMROptionNumb
line 1885:            IF (RPROG(IOPTION).EQ.IPROG) THEN ! only for options assigned to pr.
line 1886:                IF(DSMROptionFyr(IOPTION).LE.CURIYR+BASEYR) THEN !avail. next year?
line 1887:                    ! Check if the option is applicable to the current region
line 1888:                    DO i=1,RrLIST(DSMROptionRegion(IOPTION),1)
line 1889:                        IF(RNB.EQ.RrLIST(DSMROptionRegion(IOPTION),i+1)) GOTO 300
line 1890:                    ENDDO
line 1891:                    GOTO 400
line 1892: 300    CONTINUE
line 1893:                    j=j+1
line 1894:                    RPROGDEF(RNB,IPROG,j,1)=IOPTION
line 1895: 400    CONTINUE
line 1896:                ENDIF
line 1897:            ENDIF
line 1898:        ENDDO
line 1899:        RPROGDEFN(RNB,IPROG)=j
line 1900:    ENDDO
line 1901: C FOR COMMERCIAL PROGRAMS
line 1902:    DO IPROG=1,NCPROG ! do for each commercial program
line 1903:        j=0 ! counter of number of options accepted to a program
line 1904:        DO IOPTION=1,DSMCOptionNumb
line 1905:            IF (CPROG(IOPTION).EQ.IPROG) THEN ! only for options assigned to pr
line 1906:                IF(DSMCOptionFyr(IOPTION).LE.CURIYR+BASEYR) THEN !available next yr?
line 1907:                    ! Check if the option is applicable to the current region
line 1908:                    DO i=1,CrLIST(1,1)
line 1909:                        IF(RNB.EQ.CrLIST(DSMCOptionRegion(IOPTION),i+1)) GOTO 310
line 1910:                    ENDDO
line 1911:                    GOTO 410
line 1912: 310    CONTINUE
line 1913:                    j=j+1
line 1914:                    CPROGDEF(RNB,IPROG,j,1)=IOPTION
line 1915: 410    CONTINUE
line 1916:                ENDIF
line 1917:            ENDIF
line 1918:        ENDDO
line 1919:        CPROGDEFN(RNB,IPROG)=j

```

```

line 1920:  ENDDO
line 1921: C***** Termination of the Program/Subprogram *****
line 1922:  RETURN
line 1923:  END
line 1924:  SUBROUTINE DSMPRGL
line 1925: C***** Description of the Program/Subprogram *****
line 1926: C This subroutine develops load impact profiles for DSM programs (by LDC block)
line 1927: C-----
line 1928: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 1929: C*****
line 1930:  IMPLICIT NONE
line 1931: C***** Typing, Declaring and Initializing Constant Parameters ****
line 1932:  INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 1933:  INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 1934:  INCLUDE 'EMM$PARM' !<< EMM GLOBAL PARAMETERS
line 1935: C***** Typing and Declaring Variables *****
line 1936:  INTEGER*2 I,J,K,L,M,N ! multi-purpose pointers
line 1937:  REAL*4 DistLo(MAXHOUR) ! distribution of annual load over hours
line 1938:  REAL*8 load ! temporary variables for annual load forecast orig. appr.
line 1939:  INTEGER*2 K11 ! number of year after the first year of ECP time horizon
line 1940:  INTEGER*2 SYRix ! program start year index
line 1941:  INTEGER*4 NUMREC ! 4 bite record number on DAF-LSR-DB
line 1942:  CHARACTER*8 LSRname ! LSR name
line 1943: C***** COMMON and EQUIVALENCE *****
line 1944:  INCLUDE 'NCNTRL' !<< global NEMS variables
line 1945:  INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 1946:  INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 1947:  INCLUDE 'DSMHHELM' !<< HELM algorithm variables
line 1948:  INCLUDE 'DSMNEMSC' !<< results to be passed to the rest of NEMS
line 1949:  INCLUDE 'DSMTFECP' !<< communication with ECP
line 1950:  INCLUDE 'DSMTOefd' !<< communication with EFD
line 1951:  INCLUDE 'DSMTFEFP' !<< communication with EFP
line 1952:  INCLUDE 'DSMNERCR' !<< NERC region data
line 1953:  INCLUDE 'DSMCALDR' !<< calendar data
line 1954:  INCLUDE 'DSMOPTDB' !<< DSM options data base
line 1955:  INCLUDE 'DSMPROG' !<< DSM program data
line 1956: C***** Initializing Variables *****
line 1957: C***** Body of the Program/Subprogram *****
line 1958:  K11=K1-CURIYR
line 1959:  DO I=1,NRPROG+NCPROG
line 1960:    IF(NTECHAFFP(I).GT.0) THEN
line 1961:      DO SYRix=1,K11
line 1962:        DO M=1,nhour
line 1963:          SYLOAD(M)=0.0
line 1964:        ENDDO
line 1965:        DO J=1,NTECHAFFP(I) ! do for each technology affected by the program
line 1966:          IF (DeltaEnergy(J,SYRix,K11,I).NE.0.0) THEN !only if load is non 0
line 1967:            C Read a record from the Direct Access File
line 1968:              NUMREC=LSRindex(J,I)
line 1969:              READ(IODB,REC=NUMREC)LSRname,
line 1970:            &      (DistLo(m),m=1,nhour)
line 1971:              load=DeltaEnergy(J,SYRix,K11,I)
line 1972:              DO M=1,nhour
line 1973:                SYLOAD(M)=SYLOAD(M)+DistLo(M)*load
line 1974:              ENDDO
line 1975:            ENDIF
line 1976:          ENDDO
line 1977:        C develop LDC for a program, so the hours and segments are in same order as
line 1978:        C in system load LDC
line 1979:          CALL DSMLCP(2)
line 1980:          DO M=1,ECPnumBl
line 1981:            DSMPRIM(SYRix,K11,RNB,M,I)=BlockHeight(BlockNum(M))

```

```

line 1982:      ENDDO
line 1983:      ENDDO
line 1984:      ELSE
line 1985:      DO SYRix=1,K11
line 1986:          DO M=1,ECPhumBl
line 1987:              DSMPLIM(SYRix,K11,RNB,M,I)=0.0
line 1988:          ENDDO
line 1989:      ENDDO
line 1990:      ENDIF
line 1991:      ENDDO
line 1992: C***** Termination of the Program/Subprogram *****
line 1993:      RETURN
line 1994:      END
line 1995:      SUBROUTINE DSMQSI(TOSORT,INDEX,JNUMB)
line 1996:      IMPLICIT NONE
line 1997: C*****
line 1998: C***  TOSORT IS THE INPUT ARRAY (INTEGER*2)
line 1999: C***  INDEX IS THE ASSOCIATED ARRAY (INTEGER*2)
line 2000: C***  JNUMB IS THE NUMBER OF ELEMENTS
line 2001: C*****
line 2002: C***
line 2003:      INTEGER*2 TOPS(24),BOTTOM(24)
line 2004:      INTEGER*2 TOSORT(*),PIVOT
line 2005:      INTEGER*2 INDEX(*),PIVOTI
line 2006:      INTEGER*2 JNUMB
line 2007:      INTEGER*2 NUMBER
line 2008:      INTEGER*2 ITEMP
line 2009:      INTEGER*2 IPART
line 2010:      INTEGER*2 INIT
line 2011:      INTEGER*2 LIMDEX
line 2012:      INTEGER*2 MEDIAN
line 2013:      INTEGER*2 ITOP
line 2014:      INTEGER*2 IBOT
line 2015:      INTEGER*2 TEMP2I
line 2016:      REAL*4 TEMP2
line 2017:      INTEGER*2 MIN
line 2018:      INTEGER*2 I
line 2019: C***
line 2020: C*****
line 2021:      NUMBER = JNUMB
line 2022:      IF (NUMBER.GT.2) GO TO 75
line 2023:      IF (TOSORT(1).LE.TOSORT(2)) RETURN
line 2024:      ITEMP=TOSORT(1)
line 2025:      TOSORT(1)=TOSORT(2)
line 2026:      TOSORT(2)=ITEMP
line 2027:      ITEMP=INDEX(1)
line 2028:      INDEX(1)=INDEX(2)
line 2029:      INDEX(2)=ITEMP
line 2030:      RETURN
line 2031: 75 IPART=1
line 2032:      INIT=1
line 2033:      LIMDEX=1
line 2034:      IF (NUMBER.LT.10) GO TO 450
line 2035:      GO TO 400
line 2036: 100 MEDIAN=(IPART+NUMBER)/2
line 2037:      PIVOT=TOSORT(MEDIAN)
line 2038:      PIVOTI=INDEX(MEDIAN)
line 2039:      ITOP=IPART
line 2040:      IBOT=NUMBER
line 2041:      IF (TOSORT(IPART).GE.PIVOT) GO TO 150
line 2042:      TOSORT(MEDIAN)=TOSORT(IPART)
line 2043:      TOSORT(IPART)=PIVOT

```

```

line 2044: PIVOT=TOSORT(MEDIAN)
line 2045: INDEX(MEDIAN)=INDEX(IPART)
line 2046: INDEX(IPART)=PIVOTI
line 2047: PIVOTI=INDEX(MEDIAN)
line 2048: 150 IF (TOSORT(NUMBER).LE.PIVOT) GO TO 200
line 2049: TOSORT(MEDIAN)=TOSORT(NUMBER)
line 2050: TOSORT(NUMBER)=PIVOT
line 2051: PIVOT=TOSORT(MEDIAN)
line 2052: INDEX(MEDIAN)=INDEX(NUMBER)
line 2053: INDEX(NUMBER)=PIVOTI
line 2054: PIVOTI=INDEX(MEDIAN)
line 2055: IF (TOSORT(IPART).GE.PIVOT) GO TO 200
line 2056: TOSORT(MEDIAN)=TOSORT(IPART)
line 2057: TOSORT(IPART)=PIVOT
line 2058: PIVOT=TOSORT(MEDIAN)
line 2059: INDEX(MEDIAN)=INDEX(IPART)
line 2060: INDEX(IPART)=PIVOTI
line 2061: PIVOTI=INDEX(MEDIAN)
line 2062: 200 IBOT=IBOT-1
line 2063: IF (TOSORT(IBOT).LT.PIVOT) GO TO 200
line 2064: TEMP2=TOSORT(IBOT)
line 2065: TEMP2I=INDEX(IBOT)
line 2066: 250 ITOP=ITOP+1
line 2067: IF (TOSORT(ITOP).GT.PIVOT) GO TO 250
line 2068: IF (ITOP.GT.IBOT) GO TO 300
line 2069: TOSORT(IBOT)=TOSORT(ITOP)
line 2070: TOSORT(ITOP)=TEMP2
line 2071: INDEX(IBOT)=INDEX(ITOP)
line 2072: INDEX(ITOP)=TEMP2I
line 2073: GO TO 200
line 2074: 300 IF (IBOT-IPART.GE.NUMBER-ITOP) GO TO 350
line 2075: TOPS(LIMDEX)=IPART
line 2076: BOTTOM(LIMDEX)=IBOT
line 2077: IPART=ITOP
line 2078: GO TO 399
line 2079: 350 TOPS(LIMDEX)=ITOP
line 2080: BOTTOM(LIMDEX)=NUMBER
line 2081: NUMBER=IBOT
line 2082: 399 LIMDEX=LIMDEX+1
line 2083: 400 IF (NUMBER-IPART.GT.10) GO TO 100
line 2084: 450 MIN=IPART+1
line 2085: DO 425 I=MIN,NUMBER
line 2086: PIVOT=TOSORT(I)
line 2087: PIVOTI=INDEX(I)
line 2088: ITOP=I-1
line 2089: IF (TOSORT(ITOP).GE.PIVOT) GO TO 425
line 2090: 475 TOSORT(ITOP+1)=TOSORT(ITOP)
line 2091: INDEX(ITOP+1)=INDEX(ITOP)
line 2092: ITOP=ITOP-1
line 2093: IF (ITOP.EQ.0) GO TO 485
line 2094: IF (TOSORT(ITOP).LT.PIVOT) GO TO 475
line 2095: 485 TOSORT(ITOP+1)=PIVOT
line 2096: INDEX(ITOP+1)=PIVOTI
line 2097: 425 CONTINUE
line 2098: 500 LIMDEX=LIMDEX-1
line 2099: IF (LIMDEX.EQ.0) RETURN
line 2100: IPART=TOPS(LIMDEX)
line 2101: NUMBER=BOTTOM(LIMDEX)
line 2102: GO TO 400
line 2103: END
line 2104: SUBROUTINE DSMQSR(TOSORT,INDEX,FIRST,LAST)
line 2105: IMPLICIT NONE

```

```

line 2106: C*****
line 2107: C***  TOSORT IS THE INPUT ARRAY (REAL*4)
line 2108: C***  INDEX IS THE ASSOCIATED ARRAY (INTEGER*2)
line 2109: C***  FIRST IS THE FIRST ELEMENT OF THE RANGE TO BE SORTED
line 2110: C***  LAST IS THE LAST ELEMENT OF THE RANGE TO BE SORTED
line 2111: C*****
line 2112: C***
line 2113:   INTEGER*2 TOPS(24),BOTTOM(24)
line 2114:   REAL*4  TOSORT(*),PIVOT
line 2115:   INTEGER*2 INDEX(*),PIVOTI
line 2116:   INTEGER*2 JNUMB
line 2117:   INTEGER*2 NUMBER
line 2118:   INTEGER*2 ITEMP
line 2119:   INTEGER*2 IPART
line 2120:   INTEGER*2 LIMDEX
line 2121:   INTEGER*2 MEDIAN
line 2122:   INTEGER*2 ITOP
line 2123:   INTEGER*2 IBOT
line 2124:   INTEGER*2 TEMP2I
line 2125:   REAL*4  TEMP2
line 2126:   INTEGER*2 MIN
line 2127:   INTEGER*2 I
line 2128:   INTEGER*2 FIRST,LAST,SECOND,ZEROTH,ONE,TWO,ZERO,TEN
line 2129: C***
line 2130: C*****
line 2131:   ZERO=0
line 2132:   ONE=1
line 2133:   TWO=2
line 2134:   TEN=10
line 2135:   ZEROTH=FIRST-ONE
line 2136:   SECOND=FIRST+ONE
line 2137:   NUMBER=LAST
line 2138:   JNUMB=LAST-FIRST+1
line 2139:   IF (JNUMB.GT.TWO) GO TO 75
line 2140:   IF (TOSORT(FIRST).LE.TOSORT(SECOND)) RETURN
line 2141:   ITEMP=TOSORT(FIRST)
line 2142:   TOSORT(FIRST)=TOSORT(SECOND)
line 2143:   TOSORT(SECOND)=ITEMP
line 2144:   ITEMP=INDEX(FIRST)
line 2145:   INDEX(FIRST)=INDEX(SECOND)
line 2146:   INDEX(SECOND)=ITEMP
line 2147:   RETURN
line 2148: 75 IPART=FIRST
line 2149:   LIMDEX=ONE
line 2150:   IF (JNUMB.LT.TEN) GO TO 450
line 2151:   GO TO 400
line 2152: 100 MEDIAN=(IPART+NUMBER)/TWO
line 2153:   PIVOT=TOSORT(MEDIAN)
line 2154:   PIVOTI=INDEX(MEDIAN)
line 2155:   ITOP=IPART
line 2156:   IBOT=NUMBER
line 2157:   IF (TOSORT(IPART).GE.PIVOT) GO TO 150
line 2158:   TOSORT(MEDIAN)=TOSORT(IPART)
line 2159:   TOSORT(IPART)=PIVOT
line 2160:   PIVOT=TOSORT(MEDIAN)
line 2161:   INDEX(MEDIAN)=INDEX(IPART)
line 2162:   INDEX(IPART)=PIVOTI
line 2163:   PIVOTI=INDEX(MEDIAN)
line 2164: 150 IF (TOSORT(NUMBER).LE.PIVOT) GO TO 200
line 2165:   TOSORT(MEDIAN)=TOSORT(NUMBER)
line 2166:   TOSORT(NUMBER)=PIVOT
line 2167:   PIVOT=TOSORT(MEDIAN)

```

```

line 2168: INDEX(MEDIAN)=INDEX(NUMBER)
line 2169: INDEX(NUMBER)=PIVOTI
line 2170: PIVOTI=INDEX(MEDIAN)
line 2171: IF (TOSORT(IPART).GE.PIVOT) GO TO 200
line 2172: TOSORT(MEDIAN)=TOSORT(IPART)
line 2173: TOSORT(IPART)=PIVOT
line 2174: PIVOT=TOSORT(MEDIAN)
line 2175: INDEX(MEDIAN)=INDEX(IPART)
line 2176: INDEX(IPART)=PIVOTI
line 2177: PIVOTI=INDEX(MEDIAN)
line 2178: 200 IBOT=IBOT-1
line 2179: IF (TOSORT(IBOT).LT.PIVOT) GO TO 200
line 2180: TEMP2=TOSORT(IBOT)
line 2181: TEMP2I=INDEX(IBOT)
line 2182: 250 ITOP=ITOP+1
line 2183: IF (TOSORT(ITOP).GT.PIVOT) GO TO 250
line 2184: IF (ITOP.GT.IBOT) GO TO 300
line 2185: TOSORT(IBOT)=TOSORT(ITOP)
line 2186: TOSORT(ITOP)=TEMP2
line 2187: INDEX(IBOT)=INDEX(ITOP)
line 2188: INDEX(ITOP)=TEMP2I
line 2189: GO TO 200
line 2190: 300 IF (IBOT-IPART.GE.NUMBER-ITOP) GO TO 350
line 2191: TOPS(LIMDEX)=IPART
line 2192: BOTTOM(LIMDEX)=IBOT
line 2193: IPART=ITOP
line 2194: GO TO 399
line 2195: 350 TOPS(LIMDEX)=ITOP
line 2196: BOTTOM(LIMDEX)=NUMBER
line 2197: NUMBER=IBOT
line 2198: 399 LIMDEX=LIMDEX+ONE
line 2199: 400 IF (NUMBER-IPART.GT.TEN) GO TO 100
line 2200: 450 MIN=IPART+ONE
line 2201: DO 425 I=MIN,NUMBER
line 2202: PIVOT=TOSORT(I)
line 2203: PIVOTI=INDEX(I)
line 2204: ITOP=I-ONE
line 2205: IF (TOSORT(ITOP).GE.PIVOT) GO TO 425
line 2206: 475 TOSORT(ITOP+ONE)=TOSORT(ITOP)
line 2207: INDEX(ITOP+ONE)=INDEX(ITOP)
line 2208: ITOP=ITOP-ONE
line 2209: IF (ITOP.EQ.ZERO) GO TO 485
line 2210: IF (TOSORT(ITOP).LT.PIVOT) GO TO 475
line 2211: 485 TOSORT(ITOP+ONE)=PIVOT
line 2212: INDEX(ITOP+ONE)=PIVOTI
line 2213: 425 CONTINUE
line 2214: 500 LIMDEX=LIMDEX-ONE
line 2215: IF (LIMDEX.EQ.ZERO) RETURN
line 2216: IPART=TOPS(LIMDEX)
line 2217: NUMBER=BOTTOM(LIMDEX)
line 2218: GO TO 400
line 2219: END
line 2220: SUBROUTINE DSMREBC(opti,region,REBATE,AnnualEnSavings)
line 2221: C***** Description of the Program/Subprogram *****
line 2222: C This subroutine calculates current REBATE value for commercial option opti
line 2223: C-----
line 2224: C Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 2225: C*****
line 2226: IMPLICIT NONE
line 2227: C***** Typing, Declaring and Initializing Constant Parameters ****
line 2228: INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 2229: INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations

```



```

line 2230:    INCLUDE 'EMM$PARM' !<< EMM GLOBAL PARAMETERS
line 2231: C***** Typing, Declaring and Initializing Constant Parameters in c.
line 2232: C***** Typing and Declaring Variables *****
line 2233:    INTEGER*2 M,L,K,h,d,n
line 2234:    INTEGER*2 region ! current region index
line 2235:    INTEGER*2 ntech ! number of technologies
line 2236:    INTEGER*2 ti  ! technology index
line 2237:    REAL*4 teff  ! technology efficiency
line 2238:    REAL*4 capfac ! capacity factor
line 2239:    INTEGER*2 tdn  ! number of demand module technologies
line 2240:    INTEGER*2 opti ! DSM option index in the DSM option database
line 2241:    INTEGER*2 buildi ! DSM option building type
line 2242:    INTEGER*2 nt  ! current technology index for the option
line 2243:    REAL*4 IncrOptionCost ! incremental option cost
line 2244:    REAL*4 AnnualEnSavings ! Annual energy savings per appliance
line 2245:    REAL*4 REBATE ! rebate
line 2246:    REAL*4 OMcost ! temporary variable with incremental O and M costs per
line 2247:                ! unit of theoretical capacity
line 2248:    REAL*4 EIPrice ! electricity price
line 2249:    REAL*4 uecc ! annual input energy consumption to meet
line 2250:                ! one unit of service demand (e.g. GWhs.d./yr or BLu*a/yr)
line 2251:                ! in GWh/a
line 2252:    REAL*4 CENSUSvalues(MAXCRG) ! temporary vector with CENSUS division values
line 2253:    EXTERNAL DSMNVAL
line 2254: C***** Typing and Declaring Variables in Common Blocks *****
line 2255:    REAL*4 DSMROptLSR(MAXHOUR,MAXRDSMO)
line 2256:    REAL*4 HLOAD(MAXHOUR)
line 2257: C***** COMMON and EQUIVALENCE *****
line 2258:    INCLUDE 'NCNTRL' !<< global NEMS variables
line 2259:    INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 2260:    INCLUDE 'DSMOPTDB' !<< DSM options data base
line 2261:    INCLUDE 'DSMNEMSC' !<< results of LDSM to be passed to the rest of NEMS
line 2262:    INCLUDE 'DSMNRCR' !<< NERC region data
line 2263:    INCLUDE 'DSMCALDR' !<< calendar data
line 2264:    INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 2265:    INCLUDE 'DSMTFECF' !<< communication with ECP
line 2266:    INCLUDE 'COMPARM' !<< defines PARAMETERS: CMnumBldg,CMnumServ
line 2267:    INCLUDE 'COMVARS' !<< defines Parameter: CMnumRule
line 2268:    INCLUDE 'DSMCMTDB' !<< COMMERCIAL techn. database
line 2269:    INCLUDE 'BILDOUT' !<< supplies ECP solution variables
line 2270:    INCLUDE 'DSMEPRC' !<< SUPPLIES ELECTRICITY PRICES
line 2271:    INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 2272: C***** Initializing Variables *****
line 2273: C***** Body of the Program/Subprogram *****
line 2274:    IncrOptionCost=0.0
line 2275:    AnnualEnSavings=0.0
line 2276:    buildi=DSMCOptionBuildT(opti)
line 2277:    OMcost=0.0
line 2278: C FIRST 'FROM' TECHNOLOGIES
line 2279:    ntech=DSMCOptionFromTnum(opti)
line 2280:    DO L=1,ntech ! FOR EACH "FROM" DSM-DB TECHNOLOGY OF THE OPTION
line 2281:        ti=DSMCOptionFromTech(opti,L)
line 2282:        DO k=1,nCENSUSreg
line 2283:            CENSUSvalues(k)=TechEff(k,CtechCode(ti,1),CtechCode(ti,3),
line 2284:            & CtechCode(ti,2))
line 2285:        ENDDO
line 2286:        teff=DSMNVAL(CENSUSvalues,RNB,SEC(COM))
line 2287:        DO k=1,nCENSUSreg
line 2288:            CENSUSvalues(k)=CapacityFactor(k,buildi,CtechCode(ti,1))
line 2289:        ENDDO
line 2290:        capfac=DSMNVAL(CENSUSvalues,RNB,SEC(COM))
line 2291:        uecc=1.0/teff ! GWh/yr per GWhs.d./yr or GWyr/yr per BLuyr/yr

```

```

line 2292: IF (CtechCode(ti,1).EQ.6) THEN ! if lighting technology
line 2293:   uecc=uecc*8760 ! to recalculate into GWh/yr from GWyr/yr
line 2294: ENDIF
line 2295: IF (CtechCode(ti,1).EQ.6) THEN ! if lighting technology
line 2296:   IncrOptionCost=IncrOptionCost-TechCost(CtechCode(ti,3),
line 2297: &   CtechCode(ti,2),1)*1.0E+03/8760 !mln$ per BLuyr/yr
line 2298:   ! capital costs are supplied per unit of actual capacity
line 2299:   OMcost=OMcost-TechCost(CtechCode(ti,3),CtechCode(ti,2),2)
line 2300: &   *1.0E+03/(8760*capfac) ! OM costs are given per unit of th. capacity
line 2301:   ! finally they are in mln $ per BLuyr/yr per year
line 2302: ELSE
line 2303:   IncrOptionCost=IncrOptionCost-TechCost(CtechCode(ti,3),
line 2304: &   CtechCode(ti,2),1)*1.0E+03/(UNCONFA*8760) !mln$ per GWhs.d./yr
line 2305:   ! capital costs are supplied per unit of actual capacity
line 2306:   OMcost=OMcost-TechCost(CtechCode(ti,3),CtechCode(ti,2),2)
line 2307: &   *1.0E+03/(UNCONFA*8760*capfac)! OM costs are given per unit of th. c.
line 2308:   ! finally they are in mln $ per GWhs.d./yr
line 2309: ENDIF
line 2310: AnnualEnSavings=AnnualEnSavings+uecc !GWh/y per GWhs.d/y or BLuyr/y
line 2311: ENDDO
line 2312: C NOW FOR 'TO' TECHNOLOGIES
line 2313: ntech=DSMCOptionToTnum(opti)
line 2314: DO L=1,ntech ! FOR EACH "TO" DSM-DB TECHNOLOGY OF THE OPTION
line 2315:   ti=DSMCOptionToTech(opti,L)
line 2316:   DO k=1,nCENSUSreg
line 2317:     CENSUSvalues(k)=TechEff(k,CtechCode(ti,1),CtechCode(ti,3),
line 2318: &     CtechCode(ti,2))
line 2319:   ENDDO
line 2320:   teff=DSMNVAL(CENSUSvalues,RNB,SEC(COM))
line 2321:   DO k=1,nCENSUSreg
line 2322:     CENSUSvalues(k)=CapacityFactor(k,buildi,CtechCode(ti,1))
line 2323:   ENDDO
line 2324:   capfac=DSMNVAL(CENSUSvalues,RNB,SEC(COM))
line 2325:   uecc=1.0/teff ! GWh/yr per GWhs.d./yr or GWyr/yr per BLuyr/yr
line 2326:   IF (CtechCode(ti,1).EQ.6) THEN ! if lighting technology
line 2327:     uecc=uecc*8760 ! to recalculate into GWh/yr from GWyr/yr
line 2328:   ENDIF
line 2329:   nt=nt+1
line 2330:   IF (CtechCode(ti,1).EQ.6) THEN ! if lighting technology
line 2331:     IncrOptionCost=IncrOptionCost+TechCost(CtechCode(ti,3),
line 2332: &     CtechCode(ti,2),1)*1.0E+03/8760 !mln$ per BLuyr/yr
line 2333:     ! capital costs are supplied per unit of actual capacity
line 2334:     OMcost=OMcost+TechCost(CtechCode(ti,3),CtechCode(ti,2),2)
line 2335: &     *1.0E+03/(8760*capfac) ! OM costs are given per unit of th. capacity
line 2336:     ! finally they are in mln $ per BLuyr/yr per year
line 2337:   ELSE
line 2338:     IncrOptionCost=IncrOptionCost+TechCost(CtechCode(ti,3),
line 2339: &     CtechCode(ti,2),1)*1.0E+03/(UNCONFA*8760) !mln$ per GWhs.d./yr
line 2340:     ! capital costs are supplied per unit of actual capacity
line 2341:     OMcost=OMcost+TechCost(CtechCode(ti,3),CtechCode(ti,2),2)
line 2342: &     *1.0E+03/(UNCONFA*8760*capfac)! OM costs are given per unit of th. c.
line 2343:     ! finally they are in mln $ per GWhs.d./yr
line 2344:   ENDIF
line 2345:   AnnualEnSavings=AnnualEnSavings-uecc !GWh/yr per GWhs.d/yr or BLuyr/yr
line 2346: ENDDO
line 2347: REBATE=(IncrOptionCost-PAYBACK*(AnnualEnSavings*EPRICE(1,4,region)
line 2348: & -OMcost)) !per 1GWhs.d/yr or BLuyr/yr shifted
line 2349: IF (REBATE.LT.0.0) REBATE=0.0
line 2350: IF (AnnualEnSavings.LT.0.0) AnnualEnSavings=0.0
line 2351: C***** Termination of the Program/Subprogram *****
line 2352: RETURN
line 2353: END

```

```

line 2354: SUBROUTINE DSMREBR(opti,region,REBATE,AnnualEnSavings)
line 2355: C***** Description of the Program/Subprogram *****
line 2356: C This subroutine calculates current REBATE value for residential option opti
line 2357: C-----
line 2358: C Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 2359: C*****
line 2360: IMPLICIT NONE
line 2361: C***** Typing, Declaring and Initializing Constant Parameters ****
line 2362: INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 2363: INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 2364: INCLUDE 'EMM$PARM' !<< EMM GLOBAL PARAMETERS
line 2365: C***** Typing, Declaring and Initializing Constant Parameters in c.
line 2366: C***** Typing and Declaring Variables *****
line 2367: INTEGER*2 M,L,K,h,d,n,c
line 2368: INTEGER*2 region ! current NERC region
line 2369: INTEGER*2 ntech ! number of technologies
line 2370: INTEGER*2 ti ! technology index
line 2371: INTEGER*2 tdn ! number of demand module technologies
line 2372: INTEGER*2 opti ! DSM option index in the DSM option database
line 2373: INTEGER*2 buildi ! building index
line 2374: REAL*4 CurrUEC ! sum of UECs for one DSM-DTB technology
line 2375: REAL*4 IncrOptionCost ! incremental option cost
line 2376: REAL*4 AnnualEnSavings ! Annual energy savings per appliance
line 2377: REAL*4 REBATE ! rebate
line 2378: REAL*4 UEC ! UEC
line 2379: REAL*4 CENSUSvalues(MAXCRG) ! temporary vector with CENSUS division values
line 2380: EXTERNAL DSMNVAL
line 2381: C***** Typing and Declaring Variables in Common Blocks *****
line 2382: C***** COMMON and EQUIVALENCE *****
line 2383: INCLUDE 'NCNTRL' !<< global NEMS variables
line 2384: INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 2385: INCLUDE 'DSMOPTDB' !<< DSM options data base
line 2386: INCLUDE 'DSMPROG' !<< DSM program data
line 2387: INCLUDE 'DSMNEMSC' !<< results of LDSM to be passed to the rest of NEMS
line 2388: INCLUDE 'DSMCALDR' !<< calendar data
line 2389: INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 2390: INCLUDE 'DSMTFECF' !<< communication with ECP
line 2391: INCLUDE 'BILDOUT' !<< supplies ECP solution variables
line 2392: INCLUDE 'DSMRETDB' !<< Residential sector technology data
line 2393: INCLUDE 'DSMEUECS' !<< supplies residential UEC var.
line 2394: INCLUDE 'DSMEPRC' !<< SUPPLIES ELECTRICITY PRICES
line 2395: INCLUDE 'DSMNERCR' !<< NERC region data
line 2396: INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 2397: C***** Initializing Variables *****
line 2398: IncrOptionCost=0.0
line 2399: AnnualEnSavings=0.0
line 2400: C***** Body of the Program/Subprogram *****
line 2401: C FIRST 'FROM' TECHNOLOGIES
line 2402: ntech=DSMROptionFromTnum(opti)
line 2403: DO K=1,ntech ! FOR EACH "FROM" DSM-DB TECHNOLOGY OF THE OPTION
line 2404: ti=DSMROptionFromTech(opti,K)
line 2405: buildi=DSMROptionBuildT(opti)
line 2406: tdn=RtechDMtn(ti)
line 2407: CurrUEC=0.0
line 2408: DO L=1,tdn ! FOR EACH DEMAND MODULE TECHNOLOGY OF THE DSM-DB TECH.
line 2409: IncrOptionCost=IncrOptionCost-Cost(RtechDMG(ti,L),
line 2410: & RtechDMT(ti,L))*1.0E-6 ! costs in $/ln
line 2411: C Calculate UEC
line 2412: IF(RtechDMG(ti,L).EQ.1) THEN
line 2413: DO c=1,nCENSUSreg
line 2414: CENSUSvalues(k)=HTUEC(c,RtechDMG(ti,L),buildi)
line 2415: ENDDO

```

```

line 2416:   UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2417:   ELSE IF(RtechDMG(ti,L).EQ.2) THEN
line 2418:     DO c=1,nCENSUSreg
line 2419:       CENSUSvalues(k)=CLUUEC(c,RtechDMG(ti,L),buildi)
line 2420:     ENDDO
line 2421:     UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2422:   ELSE IF(RtechDMG(ti,L).EQ.3) THEN
line 2423:     DO c=1,nCENSUSreg
line 2424:       CENSUSvalues(k)=HWUEC(c,RtechDMG(ti,L),buildi)
line 2425:     ENDDO
line 2426:     UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2427:   ELSE IF(RtechDMG(ti,L).EQ.4) THEN
line 2428:     DO c=1,nCENSUSreg
line 2429:       CENSUSvalues(k)=RFUEC(c,buildi)
line 2430:     ENDDO
line 2431:     UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2432:   ELSE IF(RtechDMG(ti,L).EQ.5) THEN
line 2433:     DO c=1,nCENSUSreg
line 2434:       CENSUSvalues(k)=FZUEC(c,buildi)
line 2435:     ENDDO
line 2436:     UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2437:   ELSE IF(RtechDMG(ti,L).EQ.6) THEN
line 2438:     DO c=1,nCENSUSreg
line 2439:       CENSUSvalues(k)=LTUEC(c,buildi)
line 2440:     ENDDO
line 2441:     UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2442:   ELSE IF(RtechDMG(ti,L).EQ.7) THEN
line 2443:     DO c=1,nCENSUSreg
line 2444:       CENSUSvalues(k)=APPUEC(c,RtechDMG(ti,L),buildi)
line 2445:     ENDDO
line 2446:     UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2447:   ELSE IF(RtechDMG(ti,L).EQ.8) THEN
line 2448:     DO c=1,nCENSUSreg
line 2449:       CENSUSvalues(k)=CKUEC(c,RtechDMG(ti,L),buildi)
line 2450:     ENDDO
line 2451:     UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2452:   ELSE IF(RtechDMG(ti,L).EQ.9) THEN
line 2453:     DO c=1,nCENSUSreg
line 2454:       CENSUSvalues(k)=DRYUEC(c,RtechDMG(ti,L),buildi)
line 2455:     ENDDO
line 2456:     UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2457:   ELSE IF(RtechDMG(ti,L).EQ.10) THEN
line 2458:     DO c=1,nCENSUSreg
line 2459:       CENSUSvalues(k)=SHTUEC(c,RtechDMG(ti,L),buildi)
line 2460:     ENDDO
line 2461:     UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2462:   ENDIF
line 2463:   CurrUEC=CurrUEC+UEC*UNCONFAR
line 2464: ENDDO
line 2465: AnnualEnSavings=AnnualEnSavings+CurrUEC
line 2466: ENDDO
line 2467: C NOW FOR 'TO' TECHNOLOGIES
line 2468:   ntech=DSMROptionToTnum(opti)
line 2469:   DO K=1,ntech ! FOR EACH "TO" DSM-DB TECHNOLOGY OF THE OPTION
line 2470:     ti=DSMROptionTOTech(opti,K)
line 2471:     buildi=DSMROptionBuildT(opti)
line 2472:     tdn=RtechDMtn(ti)
line 2473:     CurrUEC=0
line 2474:     DO L=1,tdn ! FOR EACH DEMAND MODULE TECHNOLOGY OF THE DSM-DB TECH.
line 2475:       IncrOptionCost=IncrOptionCost+Cost(RtechDMG(ti,L),
line 2476:       & RtechDMT(ti,L))*1.0E-6 ! costs in $/ln
line 2477: C Calculate UEC

```

```

line 2478: IF(RtechDMG(ti,L).EQ.1) THEN
line 2479:   DO c=1,nCENSUSreg
line 2480:     CENSUSvalues(k)=HTUEC(c,RtechDMG(ti,L),buildi)
line 2481:   ENDDO
line 2482:   UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2483: ELSE IF(RtechDMG(ti,L).EQ.2) THEN
line 2484:   DO c=1,nCENSUSreg
line 2485:     CENSUSvalues(k)=CLUUEC(c,RtechDMG(ti,L),buildi)
line 2486:   ENDDO
line 2487:   UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2488: ELSE IF(RtechDMG(ti,L).EQ.3) THEN
line 2489:   DO c=1,nCENSUSreg
line 2490:     CENSUSvalues(k)=HWUEC(c,RtechDMG(ti,L),buildi)
line 2491:   ENDDO
line 2492:   UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2493: ELSE IF(RtechDMG(ti,L).EQ.4) THEN
line 2494:   DO c=1,nCENSUSreg
line 2495:     CENSUSvalues(k)=RFUEC(c,buildi)
line 2496:   ENDDO
line 2497:   UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2498: ELSE IF(RtechDMG(ti,L).EQ.5) THEN
line 2499:   DO c=1,nCENSUSreg
line 2500:     CENSUSvalues(k)=FZUEC(c,buildi)
line 2501:   ENDDO
line 2502:   UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2503: ELSE IF(RtechDMG(ti,L).EQ.6) THEN
line 2504:   DO c=1,nCENSUSreg
line 2505:     CENSUSvalues(k)=LTUEC(c,buildi)
line 2506:   ENDDO
line 2507:   UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2508: ELSE IF(RtechDMG(ti,L).EQ.7) THEN
line 2509:   DO c=1,nCENSUSreg
line 2510:     CENSUSvalues(k)=APPUEC(c,RtechDMG(ti,L),buildi)
line 2511:   ENDDO
line 2512:   UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2513: ELSE IF(RtechDMG(ti,L).EQ.8) THEN
line 2514:   DO c=1,nCENSUSreg
line 2515:     CENSUSvalues(k)=CKUEC(c,RtechDMG(ti,L),buildi)
line 2516:   ENDDO
line 2517:   UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2518: ELSE IF(RtechDMG(ti,L).EQ.9) THEN
line 2519:   DO c=1,nCENSUSreg
line 2520:     CENSUSvalues(k)=DRYUEC(c,RtechDMG(ti,L),buildi)
line 2521:   ENDDO
line 2522:   UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2523: ELSE IF(RtechDMG(ti,L).EQ.10) THEN
line 2524:   DO c=1,nCENSUSreg
line 2525:     CENSUSvalues(k)=SHTUEC(c,RtechDMG(ti,L),buildi)
line 2526:   ENDDO
line 2527:   UEC=DSMNVAL(CENSUSvalues,RNB,SEC(RES))
line 2528: ENDIF
line 2529:   CurrUEC=CurrUEC+UEC*UNCONFAR
line 2530: ENDDO
line 2531:   AnnualEnSavings=AnnualEnSavings-CurrUEC
line 2532: ENDDO
line 2533:   REBATE=IncrOptionCost-PAYBACK*AnnualEnSavings*EPRICE(1,4,region)
line 2534:   IF (REBATE.LT.0.0) REBATE=0.0
line 2535:   IF (AnnualEnSavings.LT.0.0) AnnualEnSavings=0.0
line 2536: C***** Termination of the Program/Subprogram *****
line 2537:   RETURN
line 2538:   END
line 2539: SUBROUTINE DSMRESD

```

```

line 2540: C***** Description of the Program/Subprogram *****
line 2541: C This subroutine repeats hardwired data definitions from RES demand module
line 2542: C-----
line 2543: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 2544: C*****
line 2545:   IMPLICIT NONE
line 2546: C***** Typing, Declaring and Initializing Constant Parameters ****
line 2547:   INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 2548:   INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 2549: C***** Typing and Declaring Variables *****
line 2550: C***** COMMON and EQUIVALENCE *****
line 2551:   INCLUDE 'DSMRETDDB' !<< Residential sector technology data
line 2552:   INCLUDE 'NCNTRL' !<< global NEMS variables
line 2553: C***** Initializing Variables *****
line 2554:   EquipLife(1,1)=20   ! Electric Furnace
line 2555:   EquipLife(1,2)=20   ! Heat Pump 1
line 2556:   EquipLife(1,3)=20   ! Heat Pump 2
line 2557:   EquipLife(1,4)=20   ! Heat Pump 3
line 2558:   EquipLife(2,1)=20   ! RAC 1
line 2559:   EquipLife(2,2)=20   ! RAC 2
line 2560:   EquipLife(2,3)=20   ! RAC 3
line 2561:   EquipLife(2,4)=20   ! CAC 1
line 2562:   EquipLife(2,5)=20   ! CAC 2
line 2563:   EquipLife(2,6)=20   ! CAC 3
line 2564:   EquipLife(2,7)=20   ! Heat Pump 1 cooling
line 2565:   EquipLife(2,8)=20   ! Heat Pump 2 cooling
line 2566:   EquipLife(2,9)=20   ! Heat Pump 3 cooling
line 2567:   EquipLife(3,4)=20   ! Water Heater 1
line 2568:   EquipLife(3,5)=20   ! Water Heater 2
line 2569:   EquipLife(4,1)=20   ! Refrigerator 1
line 2570:   EquipLife(4,2)=20   ! Refrigerator 2
line 2571:   EquipLife(4,3)=20   ! Refrigerator 3
line 2572:   EquipLife(4,4)=20   ! Refrigerator 4
line 2573:   EquipLife(4,5)=20   ! Refrigerator 5
line 2574:   EquipLife(5,1)=20   ! Freezer 1
line 2575:   EquipLife(5,2)=20   ! Freezer 2
line 2576:   EquipLife(5,3)=20   ! Freezer 3
line 2577:   Cost(1,1)=711.0/1.132 ! Electric Furnace
line 2578:   Cost(1,2)=2600.0/1.132 ! Heat Pump 1
line 2579:   Cost(1,3)=3200.0/1.132 ! Heat Pump 2
line 2580:   Cost(1,4)=4000.0/1.132 ! Heat Pump 3
line 2581:   Cost(2,1)=367.0/1.132 ! RAC 1
line 2582:   Cost(2,2)=383.0/1.132 ! RAC 2
line 2583:   Cost(2,3)=468.0/1.132 ! RAC 3
line 2584:   Cost(2,4)=1380.0/1.132 ! CAC 1
line 2585:   Cost(2,5)=1587.0/1.132 ! CAC 2
line 2586:   Cost(2,6)=1748.0/1.132 ! CAC 3
line 2587:   Cost(2,7)=0.0        ! Heat Pump 1 cooling
line 2588:   Cost(2,8)=0.0        ! Heat Pump 2 cooling
line 2589:   Cost(2,9)=0.0        ! Heat Pump 3 cooling
line 2590:   Cost(3,4)=270.0/1.132 ! Water Heater 1
line 2591:   Cost(3,5)=278.0/1.132 ! Water Heater 2
line 2592:   IF(BASEYR+CURIYR-1.LT.1993) THEN
line 2593:     Cost(4,1)=524.7/1.132 ! Refrigerator 1
line 2594:     Cost(4,2)=532.0/1.132 ! Refrigerator 2
line 2595:     Cost(4,3)=534.9/1.132 ! Refrigerator 3
line 2596:     Cost(4,4)=541.4/1.132 ! Refrigerator 4
line 2597:     Cost(4,5)=553.3/1.132 ! Refrigerator 5
line 2598:   ELSE
line 2599:     Cost(4,1)=573.0/1.132 ! Refrigerator 1
line 2600:     Cost(4,2)=634.0/1.132 ! Refrigerator 2
line 2601:     Cost(4,3)=654.0/1.132 ! Refrigerator 3

```

```

line 2602:      Cost(4,4)=674.0/1.132   ! Refrigerator 4
line 2603:      Cost(4,5)=736.8/1.132   ! Refrigerator 5
line 2604:      ENDIF
line 2605:      IF(BASEYR+CURIYR-1.LT.1994) THEN
line 2606:          Cost(5,1)=390.0/1.132   ! Freezer 1
line 2607:          Cost(5,2)=428.0/1.132   ! Freezer 2
line 2608:          Cost(5,3)=459.0/1.132   ! Freezer 3
line 2609:      ELSE
line 2610:          Cost(5,1)=300.0/1.132   ! Freezer 1
line 2611:          Cost(5,2)=500.0/1.132   ! Freezer 2
line 2612:          Cost(5,3)=700.0/1.132   ! Freezer 3
line 2613:      ENDIF
line 2614: C***** Body of the Program/Subprogram *****
line 2615: C***** Termination of the Program/Subprogram *****
line 2616:      RETURN
line 2617:      END
line 2618:      FUNCTION DSMRMS(PAYBACK,SECTOR)
line 2619: C***** Description of the Program/Subprogram *****
line 2620: C This subroutine calculates a after-DSM-option-application market
line 2621: C share of the TO technology expressed as a fraction of TO+FROM market share
line 2622: C-----
line 2623: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 2624: C*****
line 2625:      IMPLICIT NONE
line 2626: C***** Typing, Declaring and Initializing Constant Parameters ****
line 2627: C***** Typing and Declaring Variables *****
line 2628:      INTEGER*2 PAYBACK ! DSM option pay-back period
line 2629:      INTEGER*4 SECTOR ! SECTOR IDENTIFIER
line 2630:      REAL*4 DSMRMS ! resulting MARKET SHARE
line 2631: C***** COMMON and EQUIVALENCE *****
line 2632:      INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 2633: C***** Initializing Variables *****
line 2634: C***** Body of the Program/Subprogram *****
line 2635:      IF (SECTOR.EQ.1) THEN
line 2636:          DSMRMS=0.7
line 2637:      ELSE IF (SECTOR.EQ.2) THEN
line 2638:          DSMRMS=0.7
line 2639:      ELSE
line 2640:          GOTO 999
line 2641:      ENDIF
line 2642: C***** Termination of the Program/Subprogram *****
line 2643:      RETURN
line 2644: 999  WRITE(IMG,*)'<) Message from FUNCTION DSMRMS'
line 2645:      WRITE(IMG,*)'<)) WRONG SECTOR IDENTIFIER USED: ',SECTOR
line 2646:      WRITE(IMG,*)'<)) DSMRMS:=1.0 ANYWAY'
line 2647:      RETURN
line 2648:      END
line 2649:      SUBROUTINE DSMRST(WHOOPS)
line 2650: C***** Description of the Program/Subprogram *****
line 2651: C This subroutine reads a structure file which defines structure of
line 2652: C the analyzed system, and supplies control data for the run.
line 2653: C-----
line 2654: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 2655: C*****
line 2656:      IMPLICIT NONE
line 2657: C***** Typing, Declaring and Initializing Constant Parameters ****
line 2658:      INTEGER*2 ONE
line 2659:      PARAMETER(ONE=1)
line 2660:      INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 2661:      INCLUDE 'PARAMETR' !<< NEMS PARAMETER declarations
line 2662:      INCLUDE 'EMM$PARM' !<< EMM GLOBAL PARAMETERS
line 2663: C***** Typing and Declaring Variables *****

```

```

line 2664: INTEGER*2 I,J,K,m,n,h,t,p,r,s,v ! temporary variables
line 2665: INTEGER*4 I
line 2666: INTEGER*4 FILE ! I/O unit for struct file
line 2667: INTEGER*4 DAFnr ! DAF-LSR-DB number of records
line 2668: INTEGER*4 RRFnr ! RESIDENTIAL RESTART FILE NUMBER OF RECORDS
line 2669: INTEGER*4 CRFnr ! COMMERCIAL RESTART FILE NUMBER OF RECORDS
line 2670: CHARACTER*8 dafil ! DAF-LSR-DB file name
line 2671: CHARACTER*8 LSRnam,LSRnames(MAXCBT)! current lsr name(s)
line 2672: CHARACTER*8 DbLSRname(MAXREC) ! names of all lsrs on DAF-LSR-DB
line 2673: CHARACTER*8 REG !CURRENT NERC REGION NAME
line 2674: CHARACTER*8 LREG(MAXRLST) !CURRENT LIST OF REGION NAMES
line 2675: CHARACTER*40 DUMMYchar ! dummy character variable
line 2676: INTEGER*2 num ! number of items following the line
line 2677: LOGICAL*1 FlagEU ! becomes FALSE when first e-u for a sector is detected
line 2678: CHARACTER*12 MonNam(MAXMON) ! month names used to define LDC's
line 2679: CHARACTER*12 DtpNam(MAXDTP) ! day-type names used to define ECP LDC's
line 2680: EXTERNAL DSMCMP ! subroutine for comparing character strings
line 2681: LOGICAL DSMCMP ! as above
line 2682: INTEGER*2 ECPsgDmonth(MAXMON) ! months numbers in a ECP LDC segments
line 2683: INTEGER*2 EFDseDnM(MAXEFDS) ! Number of months in EFD LDC seasons
line 2684: INTEGER*2 DnD ! Number of day-types in a ECP/EFD LDC segment
line 2685: INTEGER*2 DnM ! Number of months in a ECP LDC segment
line 2686: INTEGER*2 DnH ! Number of hourly intervals in a ECP/EFD LDC segment
line 2687: INTEGER*2 EFDseDmonth(MAXEFDS,MAXMON) ! months in a EFD season definition
line 2688: INTEGER*2 Dday(MAXDTP) ! day-types in a ECP/EFD LDC segments
line 2689: INTEGER*2 Dhour(MAXITV,2) ! lower, upper bounds of hourly intervals
line 2690: INTEGER*2 Sindex ! current segment index
line 2691: INTEGER*2 ECPsgWidth(MAXECPs) ! widths of ECP LDC segments
line 2692: REAL*4 x ! temporary variable
line 2693: INTEGER*2 sgn ! current segment number
line 2694: CHARACTER*7 ProgNam ! current DSM Program name
line 2695: CHARACTER*1 RegLd ! Region list descriptor
line 2696: CHARACTER*1 DecTd ! Decision type descriptor
line 2697: CHARACTER*2 BuilTd ! Building type descriptor
line 2698: CHARACTER*7 FtechLSR(MAXFRT) ! FROM technologies LSR names
line 2699: CHARACTER*7 TtechLSR(MAXTOT) ! TO technologies LSR names
line 2700: INTEGER*4 TECHIDENT(MAXRRST) ! IDENTIFIERS ON RESTART FILE
line 2701: INTEGER*4 IDENTIF ! CURRENT TECHNOLOGY IDENTIFIER
line 2702: INTEGER*2 FCTC(NUMCTCE,MAXNTPO) ! FROM Commercial technology codes
line 2703: INTEGER*2 TCTC(NUMCTCE,MAXNTPO) ! TO Commercial technology codes
line 2704: INTEGER*2 iseason ! pointer to seasons
line 2705: INTEGER*2 isegment ! pointer to segments in EFD LDC (in a season )
line 2706: INTEGER*2 iseg !pointer to segments in EFD LDC'S
line 2707: REAL*4 BlocksNumber ! number of blocks as real value
line 2708: REAL*4 DeltaBlock(MSEGEFD) ! fractional part blocks number per segment
line 2709: INTEGER*2 nblock ! counter of blocks in EFD LDC
line 2710: INTEGER*2 SegmentNumber(MSEGEFD) ! indices of segments in EFD LDC
line 2711: INTEGER*2 nBlockInLDC ! EDF LDC block index
line 2712: REAL*4 DeltaHinSegment(TNSEEFD) ! actual widths of the blocks in EFD LDC
line 2713: INTEGER*2 ih ! hour index
line 2714: LOGICAL WHOOPS ! error flag
line 2715: CHARACTER*12 SLNM ! sector name used during reading mapping matrices
line 2716: C***** COMMON and EQUIVALENCE *****
line 2717: INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message file
line 2718: INCLUDE 'DSMFMGRD' !<< FILE_MGR variables declarations
line 2719: INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 2720: INCLUDE 'DSMNEMSC' !<< results to be passed to the rest of NEMS
line 2721: INCLUDE 'DSMTFECP' !<< communication with ECP
line 2722: INCLUDE 'DSMTOEFD' !<< communication with EFD
line 2723: INCLUDE 'DSMTFEFP' !<< communication with EFP
line 2724: INCLUDE 'DSMCALDR' !<< calendar data
line 2725: INCLUDE 'DSMNERCR' !<< NERC region data

```



```

line 2726:    INCLUDE 'DSMOPTDB' !<< DSM options data base
line 2727: C***** Initializing Variables *****
line 2728: C Define character strings which identify sectors
line 2729:     SecNam(RES)='RES'
line 2730:     SecNam(COM)='COM'
line 2731:     SecNam(IND)='IND'
line 2732:     SecNam(TRA)='TRA'
line 2733:     Neu=0
line 2734: C Actual number of regions to be processed
line 2735:     nNERCreg=MNUMNR-3
line 2736:     nCENSUSreg=MNUMCR-2
line 2737:     DO I=1,MAXSEC
line 2738:         DO J=1,2
line 2739:             EUINDEX(I,J)=0  !if a sector has no e-uses EUINDEX(I,J)=0
line 2740:         ENDDO
line 2741:     ENDDO
line 2742:     SumSegWidth=0
line 2743: C***** Body of the Program/Subprogram *****
line 2744: C* Read the DAF LSR database
line 2745: C* Number of records and calendar data from the first record
line 2746:     READ(IODB,REC=1)DAFnr,NMONTH,(NODAYS(I),I=1,NMONTH),
line 2747:     &((WEIGHT(J,K),J=1,NODAYS(K)),K=1,NMONTH),NODAYT,NOSEA,
line 2748:     &(MONTYP(I),I=1,NMONTH),(IDAYTQ(I,J),I=1,NODAYS(J)),J=1,NMONTH),
line 2749:     &((JDAYTP(I,J),i=1,NODAYS(J)),J=1,NMONTH),(SENAME(I),I=1,NOSEA),
line 2750:     &(DTNAME(I),I=1,NODAYT),(MONAME(i),i=1,NMONTH)
line 2751:     DO I=2,DAFnr
line 2752:         READ(IODB,REC=I)DbLSRname(I)
line 2753:     ENDDO
line 2754: C Open the structure file
line 2755:     NEW=.FALSE.
line 2756:     fname='LDSMSTR'
line 2757:     FILE=FILE_MGR('O',fname,NEW)
line 2758:     CALL DSMSKP(FILE,WHOOPS)
line 2759:     IF(WHOOPS) RETURN
line 2760: C Read number of top coincident peak hours that is to be used to calculate
line 2761: C average coincident peak load for EFP (PCP purposes)
line 2762:     READ(FILE,*)NpeakH
line 2763: C Read definition of LDCs that are to be produced by the module
line 2764: C Read definition of LDC for ECP module
line 2765: C Read number of segments
line 2766:     READ(FILE,*)ECPnumSg
line 2767: C Read definition of blocks. All hours are allocated to a number of
line 2768: C segments defined by calendar file months, day-types and hourly intervals.
line 2769: C Within each of segments blocks are defined by percentages of hours which they
line 2770: C cover. Each block can be either 'peak' or 'non-peak' type. Peak block
line 2771: C heights are represented by the peak load in the block. Non-peak blocks are
line 2772: C represented by the average load for the block.
line 2773: C DnM stores number of months, DnD - number of day-types
line 2774: C DnH - number of hourly intervals, ECPsgDnB(MAXECPS) - number of blocks
line 2775: C in a segment, ECPsgDmonth(MAXMON) - month numbers, Dday(MAXDTP)- day-type
line 2776: C numbers,Dhour(MAXITV,1) and Dhour(MAXINTV,2)
line 2777: C - lower and upper bounds of hour intervals, ECPsgDblock(MAXECPS,MAXECPB) -
line 2778: C widths of blocks defined as %% of total number of hours in a segment.
line 2779:     DO I=1,ECPnumSg ! do for all segments
line 2780: C Read segment definition
line 2781:     READ(FILE,*)n,DnM,(MonNam(J),J=1,DnM)
line 2782:     & ,DnD,(DtpNam(J),J=1,DnD)
line 2783:     & ,DnH,((Dhour(J,K),K=1,2),J=1,DnH)
line 2784:     & ,ECPsgDnB(I),(ECPsgDblock(I,J),ECPsgDbltyp(I,J)
line 2785:     & ,J=1,ECPsgDnB(I))
line 2786:     DO J=1,DnM
line 2787: C Translate names of months and day-types into their index numbers

```

```

line 2788:      DO K=1,NMONTH
line 2789:      IF(DSMCMP(MonNam(J),MONAME(K))) GOTO 1200
line 2790:      ENDDO
line 2791:      GOTO 996
line 2792: 1200  ECPsgDmonth(J)=K
line 2793:      ENDDO
line 2794:      DO J=1,DnD
line 2795:      DO K=1,NODAYT
line 2796:      IF(DSMCMP(DtpNam(J),DTNAME(K))) GOTO 1300
line 2797:      ENDDO
line 2798:      GOTO 995
line 2799: 1300  Dday(J)=K
line 2800:      ENDDO
line 2801: C For each calendar hour that is assigned to the current segment,
line 2802: C input to the ECPsgNum(MAXHOUR) table an index of the current segment, I
line 2803:      t=0 ! counts number of real hours in each of the segments (total widths)
line 2804:      DO J=1,DnM ! do for months assigned to current segment
line 2805:      l=0 ! hour in a calendar year
line 2806:      DO K=1,ECPsgDmonth(J)-1 !calculate number of last calendar hour
line 2807:      l=1+NODAYS(K)*24 !in preceding month
line 2808:      ENDDO
line 2809:      DO K=1,NODAYS(ECPsgDmonth(J)) !do for each day in a current month
line 2810:      DO p=1,DnD !check if its day-type fits to the current block
line 2811:      IF(JDAYTP(K,ECPsgDmonth(J)).EQ.Dday(p)) THEN
line 2812:      DO n=1,DnH
line 2813:      DO h=Dhour(n,1),Dhour(n,2)
line 2814:      ECPsgNum(l+h)=I
line 2815:      t=t+WEIGHT(K,ECPsgDmonth(J))
line 2816:      ENDDO
line 2817:      ENDDO
line 2818:      ENDIF
line 2819:      ENDDO
line 2820:      l=l+24
line 2821:      ENDDO
line 2822:      ENDDO
line 2823:      ECPsgWidth(I)=t
line 2824:      SumSegWidth=SumSegWidth+t
line 2825:      ENDDO
line 2826:      IF(SumSegWidth.NE.8760 .AND. SumSegWidth.NE.8784) GOTO 994
line 2827: C Now, table ECPsgNum stores, for each calendar hour, an index of an ECP LDC
line 2828: C segment to which this hour belongs to.
line 2829: C Develop a vector of ordinal numbers of hours
line 2830:      nhour=0
line 2831:      nhourr=0
line 2832:      DO I=1,NMONTH
line 2833:      DO J=1,NODAYS(I)
line 2834:      DO h=1,24
line 2835:      nhourr=nhourr+WEIGHT(J,I)
line 2836:      nhour=nhour+1
line 2837:      HourNumber(nhour)=nhour
line 2838:      HourNumberEFD(nhour)=nhour
line 2839:      HourlyWeights(nhour)=WEIGHT(J,I)
line 2840:      ENDDO
line 2841:      ENDDO
line 2842:      ENDDO
line 2843: C Sort calendar hours by segment
line 2844:      CALL DSMQSI(ECPsgNum,HourNumber,nhour)
line 2845: C
line 2846: C Calculate division points (x coordinates) that define the LDC blocks within
line 2847: C each of segments (beginning of segment=0.0, in real hours)
line 2848:      K=0 ! ordinal number for blocks in the whole LDC
line 2849:      DO I=ECPnumSg,1,-1

```

```

line 2850:      x=0.0
line 2851:      DO J=1,ECPsgDnB(I)
line 2852:          K=K+1
line 2853:          ECPblWidth(K)=ECPsgWidth(I)*ECPsgDblock(I,J)/100.0
line 2854:          ECPblSeg(K)=I
line 2855:          x=x+ECPblWidth(K)
line 2856:          ECPblockx(I,J)=x
line 2857:      ENDDO
line 2858:  ENDDO
line 2859:  C Find for each segment first and last hour (in new sort order)
line 2860:      sgn=ECPnumSg
line 2861:      ECPsgFh(sgn)=1
line 2862:      DO i=1,nhour
line 2863:          IF(ECPsgNum(i).NE.sgn) THEN
line 2864:              ECPsgLh(sgn)=i-1
line 2865:              sgn=sgn-1
line 2866:              ECPsgFh(sgn)=i
line 2867:          ENDIF
line 2868:      ENDDO
line 2869:      ECPsgLh(sgn)=nhour
line 2870:      CALL DSMSKP(FILE,WHOOPS)
line 2871:      IF(WHOOPS) RETURN
line 2872:  C
line 2873:  C Read definition of EFD LDCs
line 2874:  C
line 2875:  C First definition of seasons
line 2876:      READ(FILE,*)EFDnS
line 2877:      CALL DSMSKP(FILE,WHOOPS)
line 2878:      IF(WHOOPS) RETURN
line 2879:      DO I=1,EFDnS
line 2880:          READ(FILE,*)n,EFDseDnM(I),(MonNam(J),J=1,EFDseDnM(I)),
line 2881:          & EFDnumSeg(I),EFDnumBI(I)
line 2882:          DO J=1,EFDseDnM(I)
line 2883:              DO K=1,NMONTH
line 2884:                  IF(DSMCMP(MonNam(J),MONAME(K))) GOTO 3200
line 2885:              ENDDO
line 2886:          GOTO 996
line 2887: 3200      EFDseDmonth(I,J)=K
line 2888:          ENDDO
line 2889:      ENDDO
line 2890:      CALL DSMSKP(FILE,WHOOPS)
line 2891:      IF(WHOOPS) RETURN
line 2892:  C Read segments definitions for EFD LDCs
line 2893:      DO I=1,EFDnS
line 2894:          DO J=1,EFDnumSeg(I)
line 2895:              READ(FILE,*)n,m,DnD,(DtpNam(K),K=1,DnD)
line 2896:              & ,DnH,((Dhour(K,l),l=1,2),K=1,DnH)
line 2897:  C Calculate segment index
line 2898:          Sindex=0
line 2899:          DO s=1,n-1
line 2900:              Sindex=Sindex+EFDnumSeg(s)
line 2901:          ENDDO
line 2902:          Sindex=Sindex+m
line 2903:          DO K=1,DnD
line 2904:              DO l=1,NODAYT
line 2905:                  IF(DSMCMP(DtpNam(K),DTNAME(l))) GOTO 3300
line 2906:              ENDDO
line 2907:          GOTO 995
line 2908: 3300      Dday(K)=1
line 2909:          ENDDO
line 2910:  C For each calendar hour, input to the EFDsgNum(MAXHOUR) table a segment
line 2911:  C index

```

```

line 2912: DO K=1,EFDseDnM(n)! do for as many months as assigned to the season
line 2913: h=0 ! hour in a calendar year
line 2914: DO l=1,EFDseDmonth(n,K)-1 !calculate number of last calendar hour
line 2915: h=h+NODAYS(l)*24 !in preceding month
line 2916: ENDDO
line 2917: DO t=1,NODAYS(EFDseDmonth(n,K)) !do for each day in a current month
line 2918: DO r=1,DnD !check if its day-type fits to the current block
line 2919: IF(JDAYTP(t,EFDseDmonth(n,K)).EQ.Dday(r)) THEN
line 2920: DO s=1,DnH
line 2921: DO v=Dhour(s,1),Dhour(s,2)
line 2922: EFDsgNum(h+v)=Sindex
line 2923: ENDDO
line 2924: ENDDO
line 2925: ENDIF
line 2926: ENDDO
line 2927: h=h+24
line 2928: ENDDO
line 2929: ENDDO
line 2930: ENDDO
line 2931: ENDDO
line 2932: C Now, table EFDsgNum stores, for each calendar hour, an index of a segment to
line 2933: C which this hour belongs to
line 2934: C Sort calendar hours by segment
line 2935: CALL DSMQSI(EFDsgNum,HourNumberEFD,nhour)
line 2936: C Find indices of first/last hour in calendar year of each segment after
line 2937: C sorting by segment
line 2938: Sindex=EFDsgNum(1)
line 2939: nFirstCalHinSeg(Sindex)=1
line 2940: DO h=2,nhour
line 2941: IF(EFDsgNum(h).NE.Sindex) THEN
line 2942: nLastCalHinSeg(Sindex)=h-1
line 2943: Sindex=EFDsgNum(h)
line 2944: nFirstCalHinSeg(Sindex)=h
line 2945: ENDIF
line 2946: ENDDO
line 2947: nLastCalHinSeg(Sindex)=nhour
line 2948: C
line 2949: C Calculate number of real hours in each season
line 2950: DO ouseason=1,EFDnS
line 2951: NhourInRealSeason(ouseason)=0
line 2952: DO J=1,EFDseDnM(ouseason)
line 2953: K=EFDseDmonth(ouseason,J)
line 2954: DO r=1,NODAYS(K)
line 2955: NhourInRealSeason(ouseason)=NhourInRealSeason(ouseason)+
line 2956: & WEIGHT(r,K)*24
line 2957: ENDDO
line 2958: ENDDO
line 2959: ENDDO
line 2960: C Calculate number of real hours in each segment
line 2961: DO Sindex=1,TNSEEFD
line 2962: NhourInRealSegment(Sindex)=0
line 2963: ENDDO
line 2964: DO h=1,nhour
line 2965: Sindex=EFDsgNum(h)
line 2966: NhourInRealSegment(Sindex)=NhourInRealSegment(Sindex)+
line 2967: & HourlyWeights(HourNumberEFD(h))
line 2968: ENDDO
line 2969: C Calculate number of blocks per segment
line 2970: iseg=0
line 2971: DO ouseason=1,EFDnS
line 2972: nblock=0
line 2973: BlocksNumber=EFDnumBl(ouseason) !to transform into real number

```

```

line 2974: C Calculate base width of blocks in a season
line 2975:     EFDLDCBW(iseason)=NhourInRealSeason(iseason)/BlocksNumber
line 2976: C Find non-integer number of blocks per segment
line 2977:     DO isegment=1,EFDnumSeg(iseason)
line 2978:         iseg=iseg+1
line 2979:         BlocksNumber=NhourInRealSegment(iseg)/EFDLDCBW(iseason)
line 2980:         NumberOfBlocksInSeg(iseg)=BlocksNumber
line 2981:         nblock=nblock+NumberOfBlocksInSeg(iseg)
line 2982:         DeltaBlock(isegment)=BlocksNumber
line 2983:     & -NumberOfBlocksInSeg(iseg)
line 2984:     SegmentNumber(isegment)=iseg
line 2985:     ENDDO
line 2986: C Sort DeltaBlock values in descending order
line 2987:     CALL DSMQSR(DeltaBlock,SegmentNumber,ONE,EFDnumSeg(iseason))
line 2988:     DO s=1,EFDnumSeg(iseason)
line 2989:         IF(nblock.EQ.EFDnumBl(iseason)) GOTO 3478
line 2990:         v=SegmentNumber(s)
line 2991:         NumberOfBlocksInSeg(v)=NumberOfBlocksInSeg(v)+1
line 2992:         nblock=nblock+1
line 2993:     ENDDO
line 2994: 3478 CONTINUE
line 2995:     ENDDO
line 2996:     numSEFD=iseg
line 2997:     nBlockInLDC=0
line 2998:     ih=0
line 2999:     iseg=numSEFD+1
line 3000:     DO iseason=EFDnS,1,-1
line 3001:         DO isegment=EFDnumSeg(iseason),1,-1
line 3002:             iseg=iseg-1
line 3003: C Find indices of fist/last hour in each segment when 8760 hours of load
line 3004: C are sorted by segment and by height in each of segments
line 3005:         ih=ih+1
line 3006:         nFirstHinSeg(iseg)=ih
line 3007:         ih=ih+NhourInRealSegment(iseg)-1
line 3008:         nLastHinSeg(iseg)=ih
line 3009: C Calculate adjusted widths of blocks in each of segments so the entire
line 3010: C segment were divided into integer number of segments
line 3011:         BlocksNumber=NumberOfBlocksInSeg(iseg)
line 3012:         DeltaHinSegment(iseg)=NhourInRealSegment(iseg)/BlocksNumber
line 3013:         nBlockInLDC=nBlockInLDC+1
line 3014: C h coordinate of first block in a segment
line 3015:         hCoordinate(nBlockInLDC)=nFirstHinSeg(iseg)-1+
line 3016:     &         DeltaHinSegment(iseg)*0.5
line 3017:         DO i=2,NumberOfBlocksInSeg(iseg)
line 3018:             nBlockInLDC=nBlockInLDC+1
line 3019: C Calculate coordinates of the rest of blocks in segment
line 3020:         hCoordinate(nBlockInLDC)=hCoordinate(nBlockInLDC-1)
line 3021:     &         +DeltaHinSegment(iseg)
line 3022:     ENDDO
line 3023:     ENDDO
line 3024:     ENDDO
line 3025: C
line 3026: C
line 3027: C Read NERC regions names, their lsr names and their t&d loss factors
line 3028: C and find adequate for the LSR's record numbers on DAF-LSR-DB
line 3029:     CALL DSMSKP(FILE,WHOOPS)
line 3030:     IF(WHOOPS) RETURN
line 3031:     DO I=1,nNERCreg
line 3032:         READ(FILE,*)NERCnam(I),LSRnam,NERCtdloss(I)
line 3033:         DO l=2,DAFnr
line 3034:             IF(LSRnam.EQ.DbLSRname(l)) THEN
line 3035:                 NERClsrN(I)=l

```

```

line 3036:      GOTO 100
line 3037:      ENDIF
line 3038:      ENDDO
line 3039:      GOTO 997 ! no match for LSR write error message and stop the program
line 3040: 100  CONTINUE
line 3041:      ENDDO
line 3042: C Read number of sectors to be analyzed by the model
line 3043:      CALL DSMSKP(FILE,WHOOPS)
line 3044:      IF(WHOOPS) RETURN
line 3045:      READ(FILE,*)NumSec
line 3046:      CALL DSMSKP(FILE,WHOOPS)
line 3047:      IF(WHOOPS) RETURN
line 3048: C Read definitions of each sector in order they appear
line 3049:      DO I=1,NumSec
line 3050:          FlagEU=.TRUE.
line 3051: C Read sector name and number of end-uses in a sector
line 3052:      CALL DSMSKP(FILE,WHOOPS)
line 3053:      IF(WHOOPS) RETURN
line 3054:      READ(FILE,*)SLNAM(I),NEUSES(I)
line 3055:      DO J=1,NumSec
line 3056:          IF (SLNAM(I)(1:3).EQ.SecNam(J)) THEN
line 3057:              SEC(J) = I
line 3058:              GOTO 30
line 3059:          ENDIF
line 3060:      ENDDO
line 3061:      GOTO 999 ! if unrecognized sector write error message and stop
line 3062: 30  CONTINUE
line 3063:      DO J=1,NEUSES(I)
line 3064: C For each end-use read its name
line 3065:      CALL DSMSKP(FILE,WHOOPS)
line 3066:      IF(WHOOPS) RETURN
line 3067:      Neu=Neu+1
line 3068:      IF(FlagEU) THEN
line 3069:          EUINDEX(I,1)=Neu
line 3070:          FlagEU=.FALSE.
line 3071:      ENDIF
line 3072:      READ(FILE,*)EUNAM(Neu)
line 3073: C Read LSR name associated with an end-use in each region and the base year load
line 3074:      CALL DSMSKP(FILE,WHOOPS)
line 3075:      IF(WHOOPS) RETURN
line 3076:      DO K=1,nNERCreg
line 3077:          READ(FILE,*)LSRnam,x
line 3078:          BaseYrLd(Neu,K)=x*UNCONFA ! convert units
line 3079:          DO l=2,DAFnr
line 3080:              IF(LSRnam.EQ.DbLSRname(l)) THEN
line 3081:                  EURECNUM(Neu,K)=l
line 3082:                  GOTO 110
line 3083:              ENDIF
line 3084:          ENDDO
line 3085:          GOTO 997 ! no match for LSR write message and stop the program
line 3086: 110  CONTINUE
line 3087:      ENDDO
line 3088:      ENDDO
line 3089:      IF(EUINDEX(I,1).GT.0) EUINDEX(I,2)=Neu
line 3090:      ENDDO
line 3091: C Read matrices used for mapping between CENSUS divisions and NERC regions and
line 3092: C vice versa.
line 3093:      DO I=1,NumSec ! do for all segments
line 3094:          CALL DSMSKP(FILE,WHOOPS)
line 3095:          IF(WHOOPS) RETURN
line 3096:          READ(FILE,*)SLNM
line 3097:          DO J=1,NumSec

```

```

line 3098:      IF (SLNM(1:3).EQ.SecNam(J)) GOTO 330
line 3099:      ENDDO
line 3100:      GOTO 998 ! if unrecognized sector write error message and RETURN
line 3101: 330  CONTINUE
line 3102:      READ(FILE,*)((MappCtoN(l,K,SEC(J)),K=1,MNUMCR),l=1,MNUMNR)
line 3103: C express the mapping values as fractions of the total for a region/census div.
line 3104:      DO l=1,nNERCreg
line 3105:          DO K=1,nCENSUSreg
line 3106:              IF(MappCtoN(l,MNUMCR,SEC(J)).LE.0.0) GOTO 897
line 3107:              MappNtoC(l,K,SEC(J))=
line 3108:      &  MappCtoN(l,K,SEC(J))/MappCtoN(l,MNUMCR,SEC(J))
line 3109:          ENDDO
line 3110:      ENDDO
line 3111:      DO l=1,nNERCreg
line 3112:          DO K=1,nCENSUSreg
line 3113:              IF(MappCtoN(MNUMNR,K,SEC(J)).LE.0.0) GOTO 896
line 3114:              if(l.eq.1) then
line 3115:                  endif
line 3116:              MappCtoN(l,K,SEC(J))=
line 3117:      &  MappCtoN(l,K,SEC(J))/MappCtoN(MNUMNR,K,SEC(J))
line 3118:          ENDDO
line 3119:      ENDDO
line 3120:  ENDDO
line 3121: C Read DSM option database
line 3122: C Read Pay-back period
line 3123:      CALL DSMSKP(FILE,WHOOPS)
line 3124:      IF(WHOOPS) RETURN
line 3125:      READ(FILE,*)PAYBACK
line 3126:      CALL DSMSKP(FILE,WHOOPS)
line 3127:      IF(WHOOPS) RETURN
line 3128: C Read Residential DSM Programs Specification
line 3129:      READ(FILE,*)NRPROG
line 3130:      CALL DSMSKP(FILE,WHOOPS)
line 3131:      IF(WHOOPS) RETURN
line 3132:      DO I=1,NRPROG
line 3133:          CALL DSMSKP(FILE,WHOOPS)
line 3134:          IF(WHOOPS) RETURN
line 3135:          READ(FILE,*)v,RprogCode(I)
line 3136:      ENDDO
line 3137:      CALL DSMSKP(FILE,WHOOPS)
line 3138:      IF(WHOOPS) RETURN
line 3139: C Read Commercial DSM Programs Specification
line 3140:      READ(FILE,*)NCPROG
line 3141:      CALL DSMSKP(FILE,WHOOPS)
line 3142:      IF(WHOOPS) RETURN
line 3143:      DO I=1,NCPROG
line 3144:          CALL DSMSKP(FILE,WHOOPS)
line 3145:          IF(WHOOPS) RETURN
line 3146:          READ(FILE,*)v,CprogCode(I)
line 3147:      ENDDO
line 3148:      CALL DSMSKP(FILE,WHOOPS)
line 3149:      IF(WHOOPS) RETURN
line 3150: C READ RESIDENTIAL TECHNOLOGIES MAP
line 3151:      READ(FILE,*)RtechNumb
line 3152:      DO I=1,RtechNumb
line 3153:          CALL DSMSKP(FILE,WHOOPS)
line 3154:          IF(WHOOPS) RETURN
line 3155:          READ(FILE,*)v,RtechCode(I),RtechDMtn(I),
line 3156:      &  (RtechDMG(I,J),J=1,RtechDMtn(I)),
line 3157:      &  (RtechDMT(I,J),J=1,RtechDMtn(i)),LSRnam
line 3158:          DO l=2,DAFnr
line 3159:              IF(LSRnam.EQ.DbLSRname(l)) THEN

```

```

line 3160:      RtechLSR(I)=1
line 3161:      GOTO 1110
line 3162:      ENDIF
line 3163:      ENDDO
line 3164:      GOTO 997 ! no match for LSR write message and stop the program
line 3165: 1110 CONTINUE
line 3166:      ENDDO
line 3167: C READ REGION GROUP DEFINITION
line 3168:      CALL DSMSKP(FILE,WHOOPS)
line 3169:      IF(WHOOPS) RETURN
line 3170:      READ(FILE,*)RRlistN
line 3171:      CALL DSMSKP(FILE,WHOOPS)
line 3172:      IF(WHOOPS) RETURN
line 3173:      DO I=1,RRlistN
line 3174:      CALL DSMSKP(FILE,WHOOPS)
line 3175:      IF(WHOOPS) RETURN
line 3176:      READ(FILE,*)RRlistID(I),RrLIST(I,1),(LREG(J),J=1,RrLIST(I,1))
line 3177:      DO K=1,RrLIST(I,1)
line 3178:      DO J=1,nNERCreg
line 3179:      IF(LREG(K).EQ.NERCnam(J)) THEN
line 3180:      RrLIST(I,K+1)=J
line 3181:      GOTO 2110
line 3182:      ENDIF
line 3183:      ENDDO
line 3184:      GOTO 993
line 3185: 2110 CONTINUE
line 3186:      ENDDO
line 3187:      ENDDO
line 3188:      CALL DSMSKP(FILE,WHOOPS)
line 3189:      IF(WHOOPS) RETURN
line 3190: C READ DECISION TYPE DESCRIPTORS MAP
line 3191:      READ(FILE,*)RdecTYPn
line 3192:      CALL DSMSKP(FILE,WHOOPS)
line 3193:      IF(WHOOPS) RETURN
line 3194:      DO I=1,RdecTYPn
line 3195:      READ(FILE,*)RdecTYPid(I),RdecTYPix(I)
line 3196:      ENDDO
line 3197:      CALL DSMSKP(FILE,WHOOPS)
line 3198:      IF(WHOOPS) RETURN
line 3199: C READ BUILDING TYPE DESCRIPTORS MAP
line 3200:      READ(FILE,*)RbuildTn
line 3201:      CALL DSMSKP(FILE,WHOOPS)
line 3202:      IF(WHOOPS) RETURN
line 3203:      DO I=1,RbuildTn
line 3204:      READ(FILE,*)RbuildTid(I),RbuildTix(I)
line 3205:      ENDDO
line 3206: C READ OPTIONS DATA BASE
line 3207:      CALL DSMSKP(FILE,WHOOPS)
line 3208:      IF(WHOOPS) RETURN
line 3209:      READ(FILE,*)DSMROptionNumb
line 3210:      CALL DSMSKP(FILE,WHOOPS)
line 3211:      IF(WHOOPS) RETURN
line 3212:      DO I=1,DSMROptionNumb
line 3213:      READ(FILE,*)DSMROptionCode(I),ProgNam,RegLd,DUMMYchar,
line 3214:      & DecTd,BuilTd,DSMROptionFromTnum(I),
line 3215:      & (FtechLSR(J),J=1,DSMROptionFromTnum(I)),DSMROptionToTnum(I),
line 3216:      & (TtechLSR(J),J=1,DSMROptionToTnum(I)),DSMROptionCost(I),
line 3217:      & DSMROptionRamp(I),DSMROptionFyr(I)
line 3218:      DO J=1,NRPROG
line 3219:      IF(ProgNam.EQ.RprogCode(J)) THEN
line 3220:      RPROG(I)=J
line 3221:      GOTO 3333

```



```

line 3222:      ENDIF
line 3223:      ENDDO
line 3224:      GOTO 992
line 3225: 3333  CONTINUE
line 3226:      DO J=1,RRlistN
line 3227:      IF(RegLd.EQ.RRlistID(J)) THEN
line 3228:      DSMROptionRegion(I)=J
line 3229:      GOTO 3334
line 3230:      ENDIF
line 3231:      ENDDO
line 3232:      GOTO 991
line 3233: 3334  CONTINUE
line 3234:      DO J=1,RdecTYPn
line 3235:      IF(DecTd.EQ.RdecTYPid(J)) THEN
line 3236:      DSMROptionDecType(I)=RdecTYPix(J)
line 3237:      GOTO 3335
line 3238:      ENDIF
line 3239:      ENDDO
line 3240:      GOTO 990
line 3241: 3335  CONTINUE
line 3242:      DO J=1,RbuildTn
line 3243:      IF(BuilTd.EQ.RbuildTid(J)) THEN
line 3244:      DSMROptionBuildT(I)=RbuildTix(J)
line 3245:      GOTO 3336
line 3246:      ENDIF
line 3247:      ENDDO
line 3248:      GOTO 889
line 3249: 3336  CONTINUE
line 3250:      DO J=1,DSMROptionFromTnum(I)
line 3251:      DO I=1,RtechNumb
line 3252:      IF(FtechLSR(J).EQ.RtechCode(I)) THEN
line 3253:      DSMROptionFromTech(I,J)=1
line 3254:      GOTO 3337
line 3255:      ENDIF
line 3256:      ENDDO
line 3257:      GOTO 888
line 3258: 3337  CONTINUE
line 3259:      ENDDO
line 3260:      DO J=1,DSMROptionToTnum(I)
line 3261:      DO I=1,RtechNumb
line 3262:      IF(TtechLSR(J).EQ.RtechCode(I)) THEN
line 3263:      DSMROptionToTech(I,J)=1
line 3264:      GOTO 3338
line 3265:      ENDIF
line 3266:      ENDDO
line 3267:      GOTO 777
line 3268: 3338  CONTINUE
line 3269:      ENDDO
line 3270:      ENDDO
line 3271: C DEVELOP MAP OF RECORDS NUMBERS ON RESTART FILE FOR DIFFERENT TECHNOLOGIES
line 3272: C IN DIFFERENT REGIONS AND BUILDING TYPES
line 3273: C Read from RESIDENTIAL RESTART FILE technology identifiers
line 3274:      READ(IODB,REC=1)RRFnr
line 3275:      READ(IORR,REC=1)RRFnr
line 3276:      DO I=2,RRFnr
line 3277:      READ(IORR,REC=1)TECHIDENT(I)
line 3278:      ENDDO
line 3279:      DO I=1,RtechNumb
line 3280:      DO J=1,RtechDMtn(I)
line 3281:      DO K=1,nNERCreg
line 3282:      DO L=1,RbuildTn
line 3283:      IDENTIF=10000000*K+100000*L+1000*RtechDMG(I,J)

```

```

line 3284:  &          +10*RtechDMT(I,J)
line 3285:          DO m=2,RRFnr
line 3286:          IF(IDENTIF.EQ.TECHIDENT(m)) THEN
line 3287:            RtechRrc(I,J,K,L)=m
line 3288:            GOTO 3355
line 3289:          ENDF
line 3290:          ENDDO
line 3291:          RtechRrc(I,J,K,L)=0
line 3292: 3355    CONTINUE
line 3293:          ENDDO
line 3294:          ENDDO
line 3295:          ENDDO
line 3296:          ENDDO
line 3297:          CALL DSMSKP(FILE,WHOOPS)
line 3298:          IF(WHOOPS) RETURN
line 3299: C Read COMMERCIAL DSM OPTITONS DATA BASE *****
line 3300: C READ COMMERCIAL TECHNOLOGIES LSR MAP
line 3301:   READ(FILE,*)CtechNumb
line 3302:   DO I=1,CtechNumb
line 3303:     CALL DSMSKP(FILE,WHOOPS)
line 3304:     IF(WHOOPS) RETURN
line 3305:     READ(FILE,*)v,(CtechCode(I,J),J=1,NUMCTCE),
line 3306:     & (LSRnames(J),J=1,MAXCBT)
line 3307:     DO J=1,MAXCBT
line 3308:       DO I=2,DAFnr
line 3309:         IF(LSRnames(J).EQ.DbLSRname(I)) THEN
line 3310:           CtechLSR(I,J)=I
line 3311:           GOTO 2111
line 3312:         ENDF
line 3313:       ENDDO
line 3314:       GOTO 1997 ! no match for LSR write message and stop the program
line 3315: 2111    CONTINUE
line 3316:     ENDDO
line 3317:   ENDDO
line 3318: C READ REGION GROUP DEFINITION
line 3319:   CALL DSMSKP(FILE,WHOOPS)
line 3320:   IF(WHOOPS) RETURN
line 3321:   READ(FILE,*)CRlistN
line 3322:   CALL DSMSKP(FILE,WHOOPS)
line 3323:   IF(WHOOPS) RETURN
line 3324:   DO I=1,CRlistN
line 3325:     CALL DSMSKP(FILE,WHOOPS)
line 3326:     IF(WHOOPS) RETURN
line 3327:     READ(FILE,*)CRIlistID(I),CrLIST(I,1),(LREG(J),J=1,CrLIST(I,1))
line 3328:     DO K=1,CrLIST(I,1)
line 3329:       DO J=1,nNERCreg
line 3330:         IF(LREG(K).EQ.NERCnam(J)) THEN
line 3331:           CrLIST(I,K+1)=J
line 3332:           GOTO 2112
line 3333:         ENDF
line 3334:       ENDDO
line 3335:     GOTO 993
line 3336: 2112    CONTINUE
line 3337:   ENDDO
line 3338: ENDDO
line 3339:   CALL DSMSKP(FILE,WHOOPS)
line 3340:   IF(WHOOPS) RETURN
line 3341: C READ DECISION TYPE DESCRIPTORS MAP
line 3342:   READ(FILE,*)CdecTYPn
line 3343:   CALL DSMSKP(FILE,WHOOPS)
line 3344:   IF(WHOOPS) RETURN
line 3345:   DO I=1,CdecTYPn

```

```

line 3346:   READ(FILE,*)CdecTYPid(I),CdecTYPix(I)
line 3347:   ENDDO
line 3348:   CALL DSMSKP(FILE,WHOOPS)
line 3349:   IF(WHOOPS) RETURN
line 3350: C READ BUILDING TYPE DESCRIPTORS MAP
line 3351:   READ(FILE,*)CbuildTn
line 3352:   CALL DSMSKP(FILE,WHOOPS)
line 3353:   IF(WHOOPS) RETURN
line 3354:   DO I=1,CbuildTn
line 3355:     READ(FILE,*)CbuildTid(I),CbuildTix(I)
line 3356:   ENDDO
line 3357: C READ OPTIONS DATA BASE
line 3358:   CALL DSMSKP(FILE,WHOOPS)
line 3359:   IF(WHOOPS) RETURN
line 3360:   READ(FILE,*)DSMCOptionNumb
line 3361:   CALL DSMSKP(FILE,WHOOPS)
line 3362:   IF(WHOOPS) RETURN
line 3363:   DO I=1,DSMCOptionNumb
line 3364:     READ(FILE,*)DSMCOptionCode(I),ProgNam,RegLd,DUMMYchar,
line 3365:     & DecTd,BuilTd,DSMCOptionFromTnum(I),
line 3366:     & ((FCTC(K,J),K=1,NUMCTCE),J=1,DSMCOptionFromTnum(I)),
line 3367:     & DSMCOptionToTnum(I),
line 3368:     & ((TCTC(K,J),K=1,NUMCTCE),J=1,DSMCOptionToTnum(I)),
line 3369:     & DSMCOptionCost(I),
line 3370:     & DSMCOptionRamp(I),DSMCOptionFyr(I)
line 3371:     IF(FCTC(3,1).EQ.11 .OR. TCTC(3,1).EQ.12) GOTO 666 !for heat pumps
line 3372:     IF(FCTC(3,1).EQ.3 .AND. FCTC(3,2).NE.11) GOTO 666 !heating function
line 3373:     IF(FCTC(3,1).EQ.4 .AND. FCTC(3,2).NE.12) GOTO 666 !must go first, then
line 3374:     IF(TCTC(3,1).EQ.3 .AND. TCTC(3,2).NE.11) GOTO 666 !it has to be followed
line 3375:     IF(TCTC(3,1).EQ.4 .AND. TCTC(3,2).NE.12) GOTO 666 !by correct cooling f.
line 3376:     IF(FCTC(3,1).EQ.3 .OR. FCTC(3,1).EQ.4 .OR.
line 3377:     & TCTC(3,1).EQ.3 .OR. TCTC(3,1).EQ.4) THEN ! if heat pump option
line 3378:     IF(DSMCOptionFromTnum(I).NE.DSMCOptionToTnum(I)) GOTO 666
line 3379:     DO J=1,DSMCOptionFromTnum(I)
line 3380:       IF(FCTC(1,J).NE.TCTC(1,J)) GOTO 666 ! TO and FROM technologies have
line 3381:       ENDDO ! to be listed in the same order
line 3382:     ENDIF ! of services
line 3383:     DO J=1,NCPROG
line 3384:       IF(ProgNam.EQ.CprogCode(J)) THEN
line 3385:         CPROG(I)=J
line 3386:         GOTO 7332
line 3387:       ENDIF
line 3388:     ENDDO
line 3389:     GOTO 992
line 3390: 7332 CONTINUE
line 3391:     DO J=1,CRlistN
line 3392:       IF(RegLd.EQ.CRlistID(J)) THEN
line 3393:         DSMCOptionRegion(I)=J
line 3394:         GOTO 7334
line 3395:       ENDIF
line 3396:     ENDDO
line 3397:     GOTO 991
line 3398: 7334 CONTINUE
line 3399:     DO J=1,CdecTYPn
line 3400:       IF(DecTd.EQ.RdecTYPid(J)) THEN
line 3401:         DSMCOptionDecType(I)=CdecTYPix(J)
line 3402:         GOTO 7335
line 3403:       ENDIF
line 3404:     ENDDO
line 3405:     GOTO 990
line 3406: 7335 CONTINUE
line 3407:     DO J=1,CbuildTn

```

```

line 3408:      IF(BuilTd.EQ.CbuildTid(J)) THEN
line 3409:          DSMCOptionBuildT(I)=CbuildTix(J)
line 3410:          GOTO 7336
line 3411:      ENDIF
line 3412:      ENDDO
line 3413:      GOTO 889
line 3414: 7336  CONTINUE
line 3415:      DO J=1,DSMCOptionFromTnum(I)
line 3416:          DO I=1,CtechNumb
line 3417:              S=0
line 3418:              DO K=1,NUMCTCE
line 3419:                  IF(FCTC(K,J).EQ.CtechCode(I,K)) S=S+1
line 3420:              ENDDO
line 3421:              IF (S.EQ.NUMCTCE) THEN
line 3422:                  DSMCOptionFromTech(I,J)=1
line 3423:                  GOTO 7337
line 3424:              ENDIF
line 3425:          ENDDO
line 3426:          GOTO 888
line 3427: 7337  CONTINUE
line 3428:      ENDDO
line 3429:      DO J=1,DSMCOptionToTnum(I)
line 3430:          DO I=1,CtechNumb
line 3431:              S=0
line 3432:              DO K=1,NUMCTCE
line 3433:                  IF(TCTC(K,J).EQ.CtechCode(I,K)) S=S+1
line 3434:              ENDDO
line 3435:              IF (S.EQ.NUMCTCE) THEN
line 3436:                  DSMCOptionToTech(I,J)=1
line 3437:                  GOTO 7338
line 3438:              ENDIF
line 3439:          ENDDO
line 3440:          GOTO 777
line 3441: 7338  CONTINUE
line 3442:      ENDDO
line 3443:      ENDDO
line 3444: C DEVELOP MAP OF RECORDS NUMBERS ON RESTART FILE FOR DIFFERENT TECHNOLOGIES
line 3445: C IN DIFFERENT REGIONS AND BUILDING TYPES
line 3446: C Read from COMMERCIAL RESTART FILE technology identifiers
line 3447:      READ(IOCR,REC=1)CRFnr
line 3448:          DO I=2,CRFnr
line 3449:              READ(IOCR,REC=I)TECHIDENT(I)
line 3450:          ENDDO
line 3451:          DO I=1,CtechNumb
line 3452:              DO K=1,nNERCreg
line 3453:                  DO L=1,CbuildTn
line 3454:                      IDENTIF=1000000*K+100000*L+1000*CtechCode(I,1)
line 3455:                      &      +10*CtechCode(I,3)+CtechCode(I,2)
line 3456:                  DO m=2,CRFnr
line 3457:                      IF(IDENTIF.EQ.TECHIDENT(m)) THEN
line 3458:                          CtechRrc(I,K,L)=m
line 3459:                          GOTO 7355
line 3460:                      ENDIF
line 3461:                  ENDDO
line 3462:                  CtechRrc(I,K,L)=0
line 3463: 7355  CONTINUE
line 3464:          ENDDO
line 3465:      ENDDO
line 3466:      ENDDO
line 3467:      FILE=FILE_MGR('C',fname,NEW)
line 3468: C***** Termination of the Program/Subprogram *****
line 3469:      RETURN

```

line 3470: 999 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3471: WRITE(IMG,*)'<)) Illegal sector name ',SLNAM(I),
line 3472: &' on LDSMSTR file'
line 3473: WHOOPS=.TRUE.
line 3474: RETURN
line 3475: 998 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3476: WRITE(IMG,*)'<)) Illegal sector name ',SLNM,
line 3477: &' on LDSMSTR file'
line 3478: WHOOPS=.TRUE.
line 3479: RETURN
line 3480: 997 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3481: WRITE(IMG,*)'<)) Illegal lsr name ',LSRnam,' on LDSMSTRU file'
line 3482: WHOOPS=.TRUE.
line 3483: RETURN
line 3484: 1997 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3485: WRITE(IMG,*)'<)) Illegal lsr name ',LSRnames(J),' on LDSMSTRU'
line 3486: WHOOPS=.TRUE.
line 3487: RETURN
line 3488: 996 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3489: WRITE(IMG,*)'<)) Illegal month name ',MonNam(J),
line 3490: &' on LDSMSTR file'
line 3491: WHOOPS=.TRUE.
line 3492: RETURN
line 3493: 995 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3494: WRITE(IMG,*)'<)) Illegal day-type name ',DtpNam(J),
line 3495: &' on LDSMSTR file'
line 3496: WHOOPS=.TRUE.
line 3497: RETURN
line 3498: 994 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3499: WRITE(IMG,*)'<)) Illegal sum of widths of segments',SumSegWidth,
line 3500: &' declared for ECP LDC on LDSMSTR file'
line 3501: WHOOPS=.TRUE.
line 3502: RETURN
line 3503: 993 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3504: WRITE(IMG,*)'<)) Illegal region name ',LREG(K),
line 3505: &' on region lists used for DSM options defin. on LDSMSTR file'
line 3506: WHOOPS=.TRUE.
line 3507: RETURN
line 3508: 992 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3509: WRITE(IMG,*)'<)) Illegal DSM program name ',ProgNam,
line 3510: &' in DSM option definition on LDSMSTR file'
line 3511: WHOOPS=.TRUE.
line 3512: RETURN
line 3513: 991 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3514: WRITE(IMG,*)'<)) Illegal DSM region list descriptor ',RegLd,
line 3515: &' in DSM option definition on LDSMSTR file'
line 3516: WHOOPS=.TRUE.
line 3517: RETURN
line 3518: 990 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3519: WRITE(IMG,*)'<)) Illegal DSM decision type descriptor ',DecTd,
line 3520: &' in DSM option definition on LDSMSTR file'
line 3521: WHOOPS=.TRUE.
line 3522: RETURN
line 3523: 889 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3524: WRITE(IMG,*)'<)) Illegal building type descriptor ',BuilTd,
line 3525: &' in DSM option definition on LDSMSTR file'
line 3526: WHOOPS=.TRUE.
line 3527: RETURN
line 3528: 888 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3529: WRITE(IMG,*)'<)) Illegal FROM technology descrip.',FtechLSR(J),
line 3530: &' in DSM option definition on LDSMSTR file'
line 3531: WHOOPS=.TRUE.

```

line 3532: RETURN
line 3533: 777 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3534: WRITE(IMG,*)'<)) Illegal TO technology descriptor ',TtechLSR(J),
line 3535: &' in DSM option definition on LDSMSTR file'
line 3536: WHOOPS=.TRUE.
line 3537: RETURN
line 3538: 666 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3539: WRITE(IMG,*)'<)) In DSM option definition ',
line 3540: & DSMCOptionCode(I),' heat pump technology is wrongly defined'
line 3541: WHOOPS=.TRUE.
line 3542: RETURN
line 3543: 897 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3544: WRITE(IMG,*)'<)) 0.0 total load for NERC region #,I
line 3545: &,' in mapping matrix for sector ',SecNam(SEC(J))
line 3546: WHOOPS=.TRUE.
line 3547: RETURN
line 3548: 896 WRITE(IMG,*)'<)) Message from subroutine DSMRST'
line 3549: WRITE(IMG,*)'<)) 0.0 total load for CENSUS region #,K
line 3550: &,' in mapping matrix for sector ',SecNam(SEC(J))
line 3551: WHOOPS=.TRUE.
line 3552: RETURN
line 3553: END
line 3554: SUBROUTINE DSMSKP(file,WHOOPS)
line 3555: C***** Description of the Program/Subprogram *****
line 3556: C This subroutine reads a data line and skips the comments
line 3557: C then it positions the next read at the next line of real data
line 3558: C-----
line 3559: C Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3428
line 3560: C*****
line 3561: IMPLICIT NONE
line 3562: C***** Typing, Declaring and Initializing Constant Parameters ****
line 3563: C***** COMMON and EQUIVALENCE *****
line 3564: INCLUDE 'DSMUNITS' !<< Include file with unit number for LDSM message fil
line 3565: C***** Typing and Declaring Variables *****
line 3566: INTEGER*4 file ! Name of the file
line 3567: CHARACTER*160 line ! Current character
line 3568: CHARACTER*1 STAR ! Character variable representing a comment symbol
line 3569: LOGICAL WHOOPS ! error flag
line 3570: C***** Initializing Variables *****
line 3571: DATA STAR/'*/
line 3572: LINE=STAR
line 3573: C***** Body of the Program/Subprogram *****
line 3574: DO WHILE(LINE(1:1).EQ.STAR)
line 3575: READ(file,'(A)',END=20)line
line 3576: ENDDO
line 3577: BACKSPACE file
line 3578: C***** Termination of the Program/Subprogram *****
line 3579: RETURN
line 3580: 20 WRITE(IMG,*)'<)) Unexpected EOF - reading file: ',file
line 3581: WRITE(IMG,*)'<)) Program terminated by subroutine DSMSKP'
line 3582: WHOOPS=.TRUE.
line 3583: RETURN
line 3584: END
line 3585: SUBROUTINE DSMTOR
line 3586: C***** Description of the Program/Subprogram *****
line 3587: C This subroutine prepares data for NEMS report writer
line 3588: C-----
line 3589: C Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 3590: C*****
line 3591: IMPLICIT NONE
line 3592: C***** Typing, Declaring and Initializing Constant Parameters ****
line 3593: INCLUDE 'PARAMETR' !<< NEMS global parameters

```

```

line 3594:   INCLUDE 'EMM$PARM' !<< EMM global parameters
line 3595:   INCLUDE 'DSMDIMEN' !<< all LDSM PARAMETER declarations
line 3596: C***** Typing and Declaring Variables *****
line 3597:   INTEGER*2 SECTOR ! sector number or 0 if not to process sectorial loads
line 3598: C***** COMMON and EQUIVALENCE *****
line 3599:   INCLUDE 'NCNTRL' !<< access to NEMS global variables like
line 3600:   INCLUDE 'UDATOUT' !<< ELECTRICITY SALES VARIABLES
line 3601:   INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 3602:   INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 3603:   INCLUDE 'DSMTFEFP' !<< communication with EFP
line 3604: C***** Initializing Variables *****
line 3605: C***** Body of the Program/Subprogram *****
line 3606:   QELRSN(RNB,CURIYR)=TotSecLoad(RNB,SEC(RES))
line 3607:   QELCMN(RNB,CURIYR)=TotSecLoad(RNB,SEC(COM))
line 3608:   QELINN(RNB,CURIYR)=TotSecLoad(RNB,SEC(IND))
line 3609:   QELTRN(RNB,CURIYR)=TotSecLoad(RNB,SEC(TRA))
line 3610:   QELASN(RNB,CURIYR)=QELRSN(RNB,CURIYR) + QELCMN(RNB,CURIYR)
line 3611:   & + QELINN(RNB,CURIYR) + QELTRN(RNB,CURIYR)
line 3612: C***** Termination of the Program/Subprogram *****
line 3613:   RETURN
line 3614:   END
line 3615:   SUBROUTINE DSMTRCC(opti,PassOrFail,REBATE,uecc)
line 3616: C***** Description of the Program/Subprogram *****
line 3617: C This subroutine performs TRC test
line 3618: C TRC test is performed for years CURIYR+1 ... end of life of equipment.
line 3619: C For all the years covered by the test one single load duration curve of energy
line 3620: C savings is used. It corresponds with the CURIYR+1 ECP LDC, and CURIYR+1 UEC's.
line 3621: C Avoided costs are derived from previous year solution of ECP. Years beyond
line 3622: C the last year covered by that solution utilize data from last available year.
line 3623: C The benefits and costs are calculated per unit of theoretical capacity.
line 3624: C-----
line 3625: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 3626: C*****
line 3627:   IMPLICIT NONE
line 3628: C***** Typing, Declaring and Initializing Constant Parameters ****
line 3629:   INCLUDE 'PARAMETR' !<< NEMS global parameters
line 3630:   INCLUDE 'DSMDIMEN' !<< Include file with all PARAMETER declarations
line 3631:   INCLUDE 'EMM$PARM' !<< EMM global parameters
line 3632: C***** Typing, Declaring and Initializing Constant Parmeters in c.
line 3633:   REAL*4 PROFTHR ! Profitability threshold for the TRC test
line 3634:   PARAMETER(PROFTHR=1.0)
line 3635: C***** Typing and Declaring Variables *****
line 3636:   INTEGER*2 M,L,K,h,d,n
line 3637:   INTEGER*2 YRR
line 3638:   INTEGER*4 NUMREC ! record number
line 3639:   INTEGER*2 EqL ! minimum equipmnet lifetime
line 3640:   INTEGER*2 ntech ! number of technologies
line 3641:   INTEGER*2 ti ! technology index
line 3642:   REAL*4 teff ! technology efficiency
line 3643:   REAL*4 capfac ! capacity factor
line 3644:   INTEGER*2 tdn ! number of demand module technologies
line 3645:   INTEGER*2 opti ! DSM option index in the DSM option database
line 3646:   INTEGER*2 buildi ! DSM option building type
line 3647:   LOGICAL*1 PassOrFail ! PASS or FAIL the TRC test indicator
line 3648:   INTEGER*2 nt ! current technology index for the option
line 3649:   INTEGER*2 y ! years of equipment technical life
line 3650:   INTEGER*2 IYR ! year on the time scale used in previous year ECP run
line 3651:   REAL*4 OptionBenefits ! option benefits from the entire life of equipment
line 3652:   REAL*4 IncrOptionCost ! incremental option cost
line 3653:   REAL*4 EnSavings(MAXECTB) ! Energy savings per block of LDC
line 3654:   REAL*4 AnnualEnSavings ! Annual energy savings per appliance
line 3655:   REAL*4 OptionCost ! option costs over the entire lifetime of equipment

```

```

line 3656: REAL*4 AopB ! Annual option benefits
line 3657: REAL*4 pvfa ! factor to present value avoided costs to the beginning of pr.
line 3658: REAL*4 pvf ! present value factor
line 3659: REAL*4 pvfc ! present value factor for costs
line 3660: REAL*4 BenefitToCostRatio ! Benefits to costs ratio for the option
line 3661: INTEGER*2 StoPdist ! distance in yars between the beginning of the ECP
line 3662:           ! solution time horizon that supplies avoided costs
line 3663:           ! and the beginning of implementation of the DSM opt.
line 3664: REAL*4 REBATE ! rebate
line 3665: REAL*4 DistLo(MAXHOUR) ! distribution of load over the hours
line 3666: CHARACTER*8 LSRNAME ! LSR name
line 3667: REAL*4 OMcost ! temporary variable with incremental O and M costs per
line 3668:           ! unit of theoretical capacity
line 3669: REAL*4 EIPrice ! electricity price
line 3670: REAL*4 uecc(MAXCTECH) ! annual input energy consumption to meet
line 3671:           ! one unit of service demand (e.g. GWhs.d./yr or BLu*a/yr)
line 3672:           ! in GWh/a
line 3673: REAL*4 CENSUSvalues(MAXCRG) ! temporary vector with CENSUS division values
line 3674: EXTERNAL DSMNVAL
line 3675: C***** Typing and Declaring Variables in Common Blocks *****
line 3676: REAL*4 DSMROptLSR(MAXHOUR,MAXRDSMO)
line 3677: REAL*4 HLOAD(MAXHOUR)
line 3678: C***** COMMON and EQUIVALENCE *****
line 3679: INCLUDE 'NCNTRL' !<< NEMS global variables
line 3680: INCLUDE 'DSMHLM' !<< HELM algorithm variables
line 3681: INCLUDE 'DSMOPTDB' !<< DSM option database
line 3682: INCLUDE 'DSMNEMSC' !<< communication with NEMS
line 3683: INCLUDE 'DSMNERCR' !<< EMM region data
line 3684: INCLUDE 'DSMCALDR' !<< calendar data
line 3685: INCLUDE 'DSMUNITS' !<< message output unit
line 3686: INCLUDE 'DSMTFECF' !<< communication with ECP
line 3687: INCLUDE 'COMPARM' !<< defines PARAMETERS: CMnumBldg,CMnumServ
line 3688: INCLUDE 'COMVARS' !<< defines Parameter: CMnumRule
line 3689: INCLUDE 'DSMCMTDB' !<< COMMERCIAL technology database communication
line 3690: INCLUDE 'BILDOUT' !<< ECP solution
line 3691: INCLUDE 'DSMEPRC' !<< SUPPLIES ELECTRICITY PRICES
line 3692: INCLUDE 'DSMSECTR' !<< sector specific data and other associated variables
line 3693: C***** Initializing Variables *****
line 3694: C***** Body of the Program/Subprogram *****
line 3695: C Commercial sector Service Demand units TBtu and BLu*a
line 3696: C Determine the distance in years between the beginning of the ECP solution time
line 3697: C time horizon that supplies avoided costs and the beginning of implementation
line 3698: C of the tested DSM option.
line 3699: StoPdist=2
line 3700: C Develop a chronological hourly curve of energy savings for the DSM option
line 3701: C for CURIYR+1
line 3702: DO M=1,nhour
line 3703: SYLOAD(M)=0.0
line 3704: ENDDO
line 3705: buildi=DSMCOptionBuildT(opti)
line 3706: nt=0
line 3707: IncrOptionCost=0.0
line 3708: OMcost=0.0
line 3709: EqL=1000
line 3710: AnnualEnSavings=0.0
line 3711: C FIRST 'FROM' TECHNOLOGIES
line 3712: ntech=DSMCOptionFromTnum(opti)
line 3713: DO L=1,ntech ! FOR EACH "FROM" DSM-DB TECHNOLOGY OF THE OPTION
line 3714: ti=DSMCOptionFromTech(opti,L)
line 3715: DO k=1,nCENSUSreg
line 3716: CENSUSvalues(k)=TechEff(k,CtechCode(ti,1),CtechCode(ti,3),
line 3717: & CtechCode(ti,2))

```



```

line 3718: ENDDO
line 3719: teff=DSMNVAL(CENSUSvalues,RNB,SEC(COM))
line 3720: DO k=1,nCENSUSreg
line 3721:   CENSUSvalues(k)=CapacityFactor(k,buildi,CtechCode(ti,1))
line 3722: ENDDO
line 3723: capfac=DSMNVAL(CENSUSvalues,RNB,SEC(COM))
line 3724: uecc(L)=1.0/teff ! GWh/yr per GWhs.d./yr or GWyr/yr per BLuYr/yr
line 3725: IF (CtechCode(ti,1).EQ.6) THEN ! if lighting technology
line 3726:   uecc(L)=uecc(L)*8760 ! to recalculate into GWh/yr from GWyr/yr
line 3727: ENDIF
line 3728: nt=nt+1
line 3729: IF (CtechCode(ti,1).EQ.6) THEN ! if lighting technology
line 3730:   IncrOptionCost=IncrOptionCost-TechCost(CtechCode(ti,3),
line 3731: &   CtechCode(ti,2),1)*1.0E+03/8760 !mln$ per BLuYr/yr
line 3732:   ! capital costs are supplied per unit of actual capacity
line 3733:   OMcost=OMcost-TechCost(CtechCode(ti,3),CtechCode(ti,2),2)
line 3734: &   *1.0E+03/(8760*capfac) ! OM costs are given per unit of th. capacity
line 3735:   ! finally they are in mln $ per BLuYr/yr per year
line 3736: ELSE
line 3737:   IncrOptionCost=IncrOptionCost-TechCost(CtechCode(ti,3),
line 3738: &   CtechCode(ti,2),1)*1.0E+03/(UNCONFA*8760) !mln$ per GWhs.d./yr
line 3739:   ! capital costs are supplied per unit of actual capacity
line 3740:   OMcost=OMcost-TechCost(CtechCode(ti,3),CtechCode(ti,2),2)
line 3741: &   *1.0E+03/(UNCONFA*8760*capfac)! OM costs are given per unit of th. c.
line 3742:   ! finally they are in mln $ per GWhs.d./yr
line 3743: ENDIF
line 3744: IF(TechLife(CtechCode(ti,3),CtechCode(ti,2)).LT.EqL)
line 3745: &   EqL=TechLife(CtechCode(ti,3),CtechCode(ti,2))
line 3746:   AnnualEnSavings=AnnualEnSavings+uecc(L) !GWh/y per GWhs.d/y or BLuYr/y
line 3747:   NUMREC=CtechLSR(ti,buildi)
line 3748:   READ(IODB,REC=NUMREC)LSRname,(DistLo(h),h=1,nhour)
line 3749:   DO h=1,nhour
line 3750:     SYLOAD(h)=SYLOAD(h)+DistLo(h)*uecc(L)
line 3751: ENDDO
line 3752: ENDDO
line 3753: C NOW FOR 'TO' TECHNOLOGIES
line 3754: ntech=DSMCOptionToTnum(opti)
line 3755: DO L=1,ntech ! FOR EACH "TO" DSM-DB TECHNOLOGY OF THE OPTION
line 3756:   ti=DSMCOptionToTech(opti,L)
line 3757:   DO k=1,nCENSUSreg
line 3758:     CENSUSvalues(k)=TechEff(k,CtechCode(ti,1),CtechCode(ti,3),
line 3759: &     CtechCode(ti,2))
line 3760:   ENDDO
line 3761:   teff=DSMNVAL(CENSUSvalues,RNB,SEC(COM))
line 3762:   DO k=1,nCENSUSreg
line 3763:     CENSUSvalues(k)=CapacityFactor(k,buildi,CtechCode(ti,1))
line 3764:   ENDDO
line 3765:   capfac=DSMNVAL(CENSUSvalues,RNB,SEC(COM))
line 3766:   uecc(L)=1.0/teff ! GWh/yr per GWhs.d./yr or GWyr/yr per BLuYr/yr
line 3767:   IF (CtechCode(ti,1).EQ.6) THEN ! if lighting technology
line 3768:     uecc(L)=uecc(L)*8760 ! to recalculate into GWh/yr from GWyr/yr
line 3769:   ENDIF
line 3770:   nt=nt+1
line 3771:   IF (CtechCode(ti,1).EQ.6) THEN ! if lighting technology
line 3772:     IncrOptionCost=IncrOptionCost+TechCost(CtechCode(ti,3),
line 3773: &     CtechCode(ti,2),1)*1.0E+03/8760 !mln$ per BLuYr/yr
line 3774:     ! capital costs are supplied per unit of actual capacity
line 3775:     OMcost=OMcost+TechCost(CtechCode(ti,3),CtechCode(ti,2),2)
line 3776: &     *1.0E+03/(8760*capfac) ! OM costs are given per unit of th. capacity
line 3777:     ! finally they are in mln $ per BLuYr/yr per year
line 3778:   ELSE
line 3779:     IncrOptionCost=IncrOptionCost+TechCost(CtechCode(ti,3),

```

```

line 3780: & CtechCode(ti,2,1)*1.0E+03/(UNCONFA*8760) !mln$ per GWhs.d./yr
line 3781: ! capital costs are supplied per unit of actual capacity
line 3782: OMcost=OMcost+TechCost(CtechCode(ti,3),CtechCode(ti,2),2)
line 3783: & *1.0E+03/(UNCONFA*8760*capfac)! OM costs are given per unit of th. c.
line 3784: ! finally they are in mln $ per GWhs.d./yr
line 3785: ENDDIF
line 3786: IF(TechLife(CtechCode(ti,3),CtechCode(ti,2)).LT.EqL)
line 3787: & EqL=TechLife(CtechCode(ti,3),CtechCode(ti,2))
line 3788: AnnualEnSavings=AnnualEnSavings-uecc(L) !GWh/yr per GWhs.d/yr or BLuylr/y
line 3789: NUMREC=CtechLSR(ti,buildi)
line 3790: READ(IODB,REC=NUMREC)LSRname,(DistLo(h),h=1,nhour)
line 3791: DO h=1,nhour
line 3792: SYLOAD(h)=SYLOAD(h)-DistLo(h)*uecc(L)
line 3793: ENDDO
line 3794: ENDDO
line 3795: IF(AnnualEnSavings.LT.0.0) THEN
line 3796: PassOrFail=.FALSE.
line 3797: REBATE=2
line 3798: RETURN
line 3799: ENDDIF
line 3800: C develop LDC for an option, so the hours and segments are in same order as
line 3801: C in system load LDC
line 3802: CALL DSMLCP(2)
line 3803: DO M=1,ECPnumBl
line 3804: EnSavings(M)=BlockHeight(BlockNum(M)) ! in GW per GWhs.d/yr
line 3805: ENDDO !or BLuylr/yr
line 3806: C Calculate option benefits and costs during the entire technical life of
line 3807: C equipment (use the shortest lifetime of affected types of equipment)
line 3808: OptionBenefits=0.0
line 3809: OptionCost=IncrOptionCost ! mln $ per IGWhs.d./yr
line 3810: pvfa=(1.0+DISCFA)**StoPdist
line 3811: C IF CURIYR=1, a CURIYR ECP solution from previous run of NEMS has to be used,
line 3812: C so the beginning of the tested DSM option implementation is only one year
line 3813: C distant from the beginning of the ECP planing horizon for that solution
line 3814: C (for other CURIYR values it is always 2 year distance)
line 3815: pvf=1.0
line 3816: pvfc=1.0
line 3817: DO y=1,EqL
line 3818: pvfc=pvfc/(1.0+DISCFA)
line 3819: C We have available previous year solution of ECP which covers years
line 3820: C CURIYR-1...CURIYR+ECPSXPH-2 from which only years CURIYR+1...CURIYR+ECPSXPH-2
line 3821: C are of use for us. Remaining years of equipment lifetime utilize the marginal
line 3822: C cost data from the last available year in the ECP solution. If CURIYR=1
line 3823: C it differs appropriately.
line 3824: AopB=0.0
line 3825: IF (y.LE.ECPSXPH-StoPdist) THEN
line 3826: IYR=y+StoPdist
line 3827: C Marginal costs on EPAVOID are present valued to the beginning of the planing
line 3828: C horizon - assume that it means - to the beginning of ECP planning horizon not
line 3829: C NEMS time horizon. To present value them to the beginning of the DSM program
line 3830: C start they have to be multiplied by (1.0+DISCFA)**2
line 3831: DO M=1,ECPnumBl
line 3832: AopB=AopB+EnSavings(M)*EPAVOID(M,IYR)*pvfa
line 3833: ENDDO
line 3834: C Add avoided costs related to the total installed capacity requirement
line 3835: AopB=AopB+EnSavings(1)*EPRMRGN(IYR)*pvfa
line 3836: ELSE
line 3837: pvf=pvf/(1.0+DISCFA)
line 3838: IYR=ECPSXPH
line 3839: DO M=1,ECPnumBl
line 3840: AopB=AopB+EnSavings(M)*EPAVOID(M,IYR)*pvfa*pvf
line 3841: ENDDO

```

```

line 3842: C Add avoided costs related to the total installed capacity requirement
line 3843:   AopB=AopB+EnSavings(1)*EPRMRGN(IYR)*pvfa*pvf
line 3844:   ENDIF
line 3845:   OptionBenefits=OptionBenefits+AopB
line 3846:   OptionCost=OptionCost+(AnnualEnSavings*DSMCOptionCost(opti)+
line 3847:   & OMcost)*pvfc !in mln$ per 1GWhs.d/yr or BLuyr/yr
line 3848:   ENDDO
line 3849: C Calculate Benefits to Costs ratio for the option
line 3850:   BenefitToCostRatio=OptionBenefits/OptionCost
line 3851:   IF(OptionBenefits.LT.0.0) THEN
line 3852:     PassOrFail=.FALSE.
line 3853:     REBATE=3
line 3854:   ELSE IF(OptionCost.LT.0.0.OR.BenefitToCostRatio.GT.PROFTHR) THEN
line 3855:     PassOrFail=.TRUE.
line 3856:     REBATE=(IncrOptionCost-PAYBACK*(AnnualEnSavings*EPRICE(1,4,RNB)
line 3857:     & -OMcost)) !per 1GWhs.d/yr or BLuyr/yr shifted
line 3858:     IF (REBATE.LT.0.0) THEN
line 3859:       PassOrFail=.FALSE.
line 3860:       REBATE=4
line 3861:     ENDIF
line 3862:   ELSE
line 3863:     PassOrFail=.FALSE.
line 3864:     REBATE=5
line 3865:   ENDIF
line 3866: C***** Termination of the Program/Subprogram *****
line 3867:   RETURN
line 3868:   END
line 3869:   SUBROUTINE DSMTRCR(opti,UEC,PassOrFail,REBATE)
line 3870: C***** Description of the Program/Subprogram *****
line 3871: C This subroutine performs TRC test
line 3872: C TRC test is performed for years CURIYR+1 ... end of life of equipment.
line 3873: C For all the years covered by the test one single load duration curve of energy
line 3874: C savings is used. It corresponds with the CURIYR+1 ECP LDC, and CURIYR+1 UEC's.
line 3875: C Avoided costs are derived from previous year solution of ECP. Years beyond
line 3876: C the last year covered by that solution utilize data from last available year.
line 3877: C-----
line 3878: C   Written by Adam Kreczko, ICF Resources, Inc., phone: (703)-934-3353
line 3879: C*****
line 3880:   IMPLICIT NONE
line 3881: C***** Typing, Declaring and Initializing Constant Parameters ****
line 3882:   INCLUDE 'PARAMETR' !<<
line 3883:   INCLUDE 'DSMDIMEN' !<< Include file with all PARAMETER declarations
line 3884:   INCLUDE 'EMM$PARM' !<<
line 3885: C***** Typing, Declaring and Initializing Constant Parmeters in c.
line 3886:   REAL*4 PROFTHR ! Profitability threshold for the TRC test
line 3887:   PARAMETER(PROFTHR=1.0)
line 3888: C***** Typing and Declaring Variables *****
line 3889:   INTEGER*2 M,L,K,h,d,n
line 3890:   INTEGER*2 YRR
line 3891:   INTEGER*4 NUMREC ! record number
line 3892:   INTEGER*2 EqL ! minimum equipmnet lifetime
line 3893:   INTEGER*2 ntech ! number of technologies
line 3894:   INTEGER*2 ti ! technology index
line 3895:   INTEGER*2 tdn ! number of demand module technologies
line 3896:   INTEGER*2 opti ! DSM option index in the DSM option database
line 3897:   LOGICAL*1 PassOrFail ! PASS or FAIL the TRC test indicator
line 3898:   INTEGER*2 ntaf ! first entry on the list of technologies affected by DSM
line 3899:   ! programs for current option
line 3900:   INTEGER*2 ntaf ! as above but last entry
line 3901:   INTEGER*2 nt ! current technology index for the option
line 3902:   REAL*4 UEC(NYRESTR,MAXDMT) ! UEC matrix, first index years, second techn.
line 3903:   REAL*4 CurrUEC ! sum of UECs for one DSM-DTB technology

```

```

line 3904: INTEGER*2 y ! years of equipment technical life
line 3905: INTEGER*2 IYR ! year on the time scale used in previous year ECP run
line 3906: REAL*4 OptionBenefits ! option benefits from the entire life of equipment
line 3907: REAL*4 IncrOptionCost ! incremental option cost
line 3908: REAL*4 EnSavings(MAXECTB) ! Energy savings per block of LDC
line 3909: REAL*4 AnnualEnSavings ! Annual energy savings per appliance
line 3910: REAL*4 OptionCost ! option costs over the entire lifetime of equipment
line 3911: REAL*4 AopB ! Annual option benefits
line 3912: REAL*4 pvfa ! factor to present value avoided costs to the beginning of pr.
line 3913: REAL*4 pvf ! present value factor
line 3914: REAL*4 pvfc ! present value factor for costs
line 3915: REAL*4 BenefitToCostRatio ! Benefits to costs ratio for the option
line 3916: INTEGER*2 StoPdist ! distance in years between the beginning of the ECP
line 3917: ! solution time horizon that supplies avoided costs
line 3918: ! and the beginning of implementation of the DSM opt.
line 3919: REAL*4 REBATE ! rebate
line 3920: REAL*4 DistLo(MAXHOUR) ! distribution of load over the hours
line 3921: CHARACTER*8 LSRNAME ! LSR name
line 3922: C***** Typing and Declaring Variables in Common Blocks *****
line 3923: REAL*4 DSMROptLSR(MAXHOUR,MAXRDSMO)
line 3924: REAL*4 HLOAD(MAXHOUR)
line 3925: C***** COMMON and EQUIVALENCE *****
line 3926: INCLUDE 'NCNTRL' !<<
line 3927: INCLUDE 'DSMHELM' !<< HELM algorithm variables
line 3928: INCLUDE 'DSMOPTDB' !<<
line 3929: INCLUDE 'DSMPROG' !<<
line 3930: INCLUDE 'DSMNEMSC' !<<
line 3931: INCLUDE 'DSMCALDR' !<<
line 3932: INCLUDE 'DSMUNITS' !<<
line 3933: INCLUDE 'DSMTFECP' !<<
line 3934: INCLUDE 'BILDOUT' !<<
line 3935: INCLUDE 'DSMRETDB' !<< Residential technology data
line 3936: INCLUDE 'DSMEPRC' !<< SUPPLIES ELECTRICITY PRICE
line 3937: C***** Initializing Variables *****
line 3938: C***** Body of the Program/Subprogram *****
line 3939: C Determine the distance in years between the beginning of the ECP solution time
line 3940: C time horizon that supplies avoided costs and the beginning of implementation
line 3941: C of the tested DSM option.
line 3942: StoPdist=2
line 3943: C Develop a chronological hourly curve of energy savings for the DSM option
line 3944: C for CURIYR+1
line 3945: DO M=1,nhour
line 3946: SYLOAD(M)=0.0
line 3947: ENDDO
line 3948: YRR=K1-1 ! UEC data cover period 1991-2015
line 3949: nt=0
line 3950: IncrOptionCost=0.0
line 3951: EqL=1000
line 3952: AnnualEnSavings=0.0
line 3953: C FIRST 'FROM' TECHNOLOGIES
line 3954: ntech=DSMROptionFromTnum(opti)
line 3955: DO K=1,ntech ! FOR EACH "FROM" DSM-DB TECHNOLOGY OF THE OPTION
line 3956: ti=DSMROptionFromTech(opti,K)
line 3957: tdn=RtechDMtn(ti)
line 3958: CurrUEC=0
line 3959: DO L=1,tdn ! FOR EACH DEMAND MODULE TECHNOLOGY OF THE DSM-DB TECH.
line 3960: nt=nt+1
line 3961: IncrOptionCost=IncrOptionCost-Cost(RtechDMG(ti,L),
line 3962: & RtechDMT(ti,L))*1.0E-6 ! costs in $/ln
line 3963: IF(EquipLife(RtechDMG(ti,L),RtechDMT(ti,L)).LT.EqL)
line 3964: & EqL=EquipLife(RtechDMG(ti,L),RtechDMT(ti,L))
line 3965: CurrUEC=CurrUEC+UEC(YRR,nt)

```

```

line 3966:      ENDDO
line 3967:      AnnualEnSavings=AnnualEnSavings+CurrUEC
line 3968:      NUMREC=RtechLSR(ti)
line 3969:      READ(IODB,REC=NUMREC)LSRname,(DistLo(h),h=1,nhour)
line 3970:      DO h=1,nhour
line 3971:          SYLOAD(h)=SYLOAD(h)+DistLo(h)*CurrUEC
line 3972:      ENDDO
line 3973:  ENDDO
line 3974: C NOW FOR 'TO' TECHNOLOGIES
line 3975:      ntech=DSMROptionToTnum(opti)
line 3976:      DO K=1,ntech ! FOR EACH "TO" DSM-DB TECHNOLOGY OF THE OPTION
line 3977:          ti=DSMROptionTOTech(opti,K)
line 3978:          tdn=RtechDMtn(ti)
line 3979:          CurrUEC=0
line 3980:          DO L=1,tdn ! FOR EACH DEMAND MODULE TECHNOLOGY OF THE DSM-DB TECH.
line 3981:              IncrOptionCost=IncrOptionCost+Cost(RtechDMG(ti,L),
line 3982:              & RtechDMT(ti,L))*1.0E-6 ! costs in $/m
line 3983:              IF(EquipLife(RtechDMG(ti,L),RtechDMT(ti,L)).LT.EqL)
line 3984:              & EqL=EquipLife(RtechDMG(ti,L),RtechDMT(ti,L))
line 3985:              nt=nt+1
line 3986:              CurrUEC=CurrUEC+UEC(YRR,nt)
line 3987:          ENDDO
line 3988:          AnnualEnSavings=AnnualEnSavings-CurrUEC
line 3989:          NUMREC=RtechLSR(ti)
line 3990:          READ(IODB,REC=NUMREC)LSRname,(DistLo(h),h=1,nhour)
line 3991:          DO h=1,nhour
line 3992:              SYLOAD(h)=SYLOAD(h)-DistLo(h)*CurrUEC
line 3993:          ENDDO
line 3994:  ENDDO
line 3995:  IF(AnnualEnSavings.LT.0.0) THEN
line 3996:      PassOrFail=.FALSE.
line 3997:      REBATE=2
line 3998:      RETURN
line 3999:  ENDIF
line 4000: C develop LDC for an option, so the hours and segments are in same order as
line 4001: C in system load LDC
line 4002:      CALL DSMLCP(2)
line 4003:      DO M=1,ECPnumBl
line 4004:          EnSavings(M)=BlockHeight(BlockNum(M))
line 4005:      ENDDO
line 4006: C Calculate option benefits and costs during the entire technical life of
line 4007: C equipment (use the shortest lifetime of affected types of equipment)
line 4008:      OptionBenefits=0.0
line 4009:      OptionCost=IncrOptionCost
line 4010:      pvfa=(1.0+DISCFA)**StoPdist
line 4011: C IF CURIYR=1, a CURIYR ECP solution from previous run of NEMS has to be used,
line 4012: C so the beginning of the tested DSM option implementation is only one year
line 4013: C distant from the beginning of the ECP planing horizon for that solution
line 4014: C (for other CURIYR values it is always 2 year distance)
line 4015:      pvf=1.0
line 4016:      pvfc=1.0
line 4017:      DO y=1,EqL
line 4018:          pvfc=pvfc/(1.0+DISCFA)
line 4019: C We have available previous year solution of ECP which covers years
line 4020: C CURIYR-1...CURIYR+ECP$XPH-2 from which only years CURIYR+1...CURIYR+ECP$XPH-2
line 4021: C are of use for us. Remaining years of equipment lifetime utilize the marginal
line 4022: C cost data from the last available year in the ECP solution.
line 4023:      AopB=0.0
line 4024:      IF (y.LE.ECP$XPH-StoPdist) THEN
line 4025:          IYR=y+StoPdist
line 4026: C Marginal costs on EPAVOID are present valued to the beginning of the planing
line 4027: C horizon - assume that it means - to the beginning of ECP planning horizon not

```

```

line 4028: C NEMS time horizon. To present value them to the beginning of the DSM program
line 4029: C start they have to be multiplied by (1.0+DISCFA)**2
line 4030:     DO M=1,ECPnumBI
line 4031:     AopB=AopB+EnSavings(M)*EPAVOID(M,IYR)*pvfa
line 4032:     ENDDO
line 4033: C Add avoided costs related to the total installed capacity requirement
line 4034:     AopB=AopB+EnSavings(1)*EPRMRGN(IYR)*pvfa
line 4035:     ELSE
line 4036:     pvf=pvf/(1.0+DISCFA)
line 4037:     IYR=ECP$XPH
line 4038:     DO M=1,ECPnumBI
line 4039:     AopB=AopB+EnSavings(M)*EPAVOID(M,IYR)*pvfa*pvf
line 4040:     ENDDO
line 4041: C Add avoided costs related to the total installed capacity requirement
line 4042:     AopB=AopB+EnSavings(1)*EPRMRGN(IYR)*pvfa*pvf
line 4043:     ENDIF
line 4044:     OptionBenefits=OptionBenefits+AopB
line 4045:     OptionCost=OptionCost+AnnualEnSavings*DSMROptionCost(opti)*pvfc
line 4046:     ENDDO
line 4047: C Calculate Benefits to Costs ratio for the option
line 4048:     BenefitToCostRatio=OptionBenefits/OptionCost
line 4049:     IF(OptionBenefits.LT.0.0) THEN
line 4050:     PassOrFail=.FALSE.
line 4051:     REBATE=3
line 4052:     ELSE IF(OptionCost.LT.0.0.OR.BenefitToCostRatio.GT.PROFTHR) THEN
line 4053:     PassOrFail=.TRUE.
line 4054:     REBATE=IncrOptionCost-PAYBACK*AnnualEnSavings*EPRICE(1,4,RNB)
line 4055:     IF (REBATE.LT.0.0) THEN
line 4056:     PassOrFail=.FALSE.
line 4057:     REBATE=4
line 4058:     ENDIF
line 4059:     ELSE
line 4060:     PassOrFail=.FALSE.
line 4061:     REBATE=5
line 4062:     ENDIF
line 4063: C***** Termination of the Program/Subprogram *****
line 4064:     RETURN
line 4065:     END
C ===== (DSMCALDR) =====
C MONTYP(MAXMON) - SEASON ASSIGNMENT OF MONTH
C JDAYTP(MAXDAY,MAXMON) - DAY-TYPE ASSIGNMENT FOR EACH DAY
C WEIGHT(MAXDAY,MAXMON) - WEIGHTS INDICATING NUMBER OF DAYS REPRESENTED !
C     BY EACH MODELLED DAY
C HourlyWeights(MAXHOUR) - HOURLY WEIGHTS FOR CALENDAR HOURS
C NODAYS(MAXMON) - NUMBER OF DAYS IN EACH MONTH !
C NODAYT - NUMBER OF DAY-TYPES
C NMONTH - NUMBER OF MONTHS !
C NOSEA - NUMBER OF SEASONS
C MONAME(MAXMON) - MONTH NAMES
C SENAME(MAXSEA) - SEASON NAMES
C DTNAME(MAXDTP) - DAY-TYPE NAMES
C IDAYTQ(MAXDAY,MAXMON) - NUMBER OF REAL DAYS WHICH ARE REPRESENTED
C     BY DAYTYPE J IN MONTH I.
C nhour - total number of hours in the calendar year
C nhouryr - total number of hours in the real year
C HourNumber(MAXHOUR) - indices of hours in the calendar year sorted for ECP
C HourNumberEFD(MAXHOUR) - indices of hours in the calendar year sorted for EFD
C HourlyWeights(MAXHOUR) - weights for hours in the calendar year
  INTEGER*2 HourlyWeights(MAXHOUR)
  INTEGER*2 HourNumber(MAXHOUR)
  INTEGER*2 HourNumberEFD(MAXHOUR)
  INTEGER*2 nhour,nhouryr,MONTYP(MAXMON),JDAYTP(MAXDAY,MAXMON),

```

```

1    WEIGHT(MAXDAY,MAXMON),NODAYS(MAXMON),
2    NODAYT,NMONTH,NOSEA,IDAYTQ(MAXDTP,MAXMON)
    CHARACTER*12 MONAME(MAXMON)
    CHARACTER*12 SENAME(MAXSEA)
    CHARACTER*12 DTNAME(MAXDTP)
    COMMON/DSMCLn/nhour,nhouryr,NOSEA,NMONTH,NODAYT,NODAYS,MONTYP,
& JDAYTP,WEIGHT,IDAYTQ,HourNumber,HourlyWeights,HourNumberEFD
    COMMON/DSMCLc/MONAME,SENAME,DTNAME
C===== (DSMCMTDB) =====
C This INCLUDE file is a part of CMTechnologyChoice subroutine from COMMERCIAL
C Demand Module. It supplies commercial technology data.
C=====
C Variables read or set from Technology Characterization Table, KTECH:

    INTEGER*4 FuelbyTech (CMnumTech,CMnumMajFl)
    ! Set to 1 if tech (row) uses fuel (col); 0 otherwise

    INTEGER*4 TechbyService (CMnumServ,CMnumTech)
    ! Set to 1 if service (row) provided by tech (col); 0 otherwise

    INTEGER*4 CMnumTechsforService (CMnumServ)
    ! For service s, CMnumTechsforService (s) is the number of
    ! technologies input from the technology characterization
    ! file, KTECH, that provide that service.

    INTEGER*4 TechsforService (CMnumServ,CMnumTech)
    ! For service s, TechsforService (s,TsubS) contains the
    ! technology subscript, t, of a technology providing
    ! service s, for each value of TsubS between 1 and
    ! CMnumTechsforService (s). The value of t is that assigned
    ! in KTECH, and will be between 1 and CMnumTech.

    ! REAL*4 TechShareofService (MNUMCR,CMnumBldg,CMnumServ,
    !                             CMnumTech,CMnumEqVint)
    ! Declared above under the heading, 'Calculated Variables.' The
    ! proportion of a given service demand that is satisfied by
    ! equipment of a particular technology & vintage within a given
    ! Census Division and building type for the previous year,
    ! until recalculated for the current year in the Technology
    ! Choice subroutine. This array is initialized for 1989
    ! using CBECS shares input from KTECH.

    REAL*4 TechEff (MNUMCR,CMnumServ,CMnumTech,CMnumEqVint)
    ! Efficiencies of specific equipment, with allowance for
    ! regional variation, and equipment use for multiple services

    REAL*4 TechCost (CMnumTech,CMnumEqVint,2)
    ! For equipment of technology t and vintage v:
    ! TechCost(t,v,1) = unit installed capital cost of equipment
    ! TechCost(t,v,2) = annual O&M cost per unit service demand

    REAL*4 TechLife (CMnumTech,CMnumEqVint)
    ! Average life expectancy of equipment, in years

    INTEGER*4 TechAvailability (CMnumTech,CMnumEqVint,2)
    ! For equipment of technology t and vintage v:
    ! TechAvailability(t,v,1) = calendar year first available to buy
    ! TechAvailability(t,v,2) = last year of equipment availability

    REAL*4 CapacityFactor (MNUMCR-2,CMnumBldg,CMnumServ)
    ! From file KCAPFAC
    ! Capacity factors by region, building type, and service

```

REAL*4 BehaviorShare (CMnumBldg,CMDecision,CMnumRule)
! From file KBEHAV;
! For technology choice decision type d in building type b,
! BehaviorShare (b,d,r) is the proportion of consumers following
! behavior rule r during the equipment selection process.

COMMON /KDAT/

* FuelbyTech, ! fuel-tech crosstab by t,f
* TechbyService, ! service-tech crosstab by s,t
* CMnumTechsforService, ! no. of techs for ea serv. by s
* TechsforService, ! subscripts of techs, ea serv by s,TsubS
* TechEff, ! equipment efficiency by r,s,t,v
* TechCost, ! capital, O&M costs by t,v
* TechLife, ! avg equipment lifetime (yrs) by t,v
* TechAvailability, ! 1st & last yr of equip avail by t,v
* BehaviorShare, ! consumer behav proportions by b,d,u
* CapacityFactor ! bldg capacity factors by r,b,s

C ===== (DSMDIMEN) =====
C This common block contains all PARAMETER statements of LDSM module
C MAXSEC - NUMBER OF SECTORS
C RES - RESIDENTIAL SECTOR INDEX
C COM - COMMERCIA SECTOR INDEX
C IND - INDUSTRIAL SECTOR INDEX
C TRA - TRANSPORTATION SECTOR INDEX
C MAXCRG - MAXIMUM NUMBER OF CENSUS REGIONS
C MAXREC - MAXIMUM NUMBER OF RECORDS ON DAF-LSR-DB
C MAXMON - MAXIMUM NUMBER OF MONTHS
C MAXSEA - MAXIMUM NUMBER OF SEASONS
C MAXDTP - MAXIMUM NUMBER OF DAY-TYPES
C MAXNRG - MAXIMUM NUMBER OF NERC REGIONS
C MAXHOUR - MAXIMUM NUMBER OF HOURS IN A YEAR +1 = MAXMON*MAXDAY*24+1
C MAXRHOUR - MAXIMUM NUMBER OF HOURS IN A REAL YEAR
C MAXDAY - MAXIMUM NUMBER OF DAYS PER MONTH
C MAXEU - MAXIMUM NUMBER OF END-USES
C NYRRD - NUMBER OF YEARS USED IN RESIDENTIAL DEMAND SUBMODULE
C NCRRD - NUMBER OF CENSUS REGIONS USED IN RESIDENTIAL DEMAND SUBMODULE
C NYRTR - NUMBER OF YEARS USED IN TRANSPORTATION SECTOR DEM. MODULE
C NCRTR - NUMBER OF CENSUS REGION USED IN TRANSPORTATION SECTOR DEM. MODULE
C NEFHTR - NUMBER OF ENERGY FORMS USED FOR RESIDENTIAL HEATING
C NEFCOO - NUMBER OF ENERGY FORMS USED FOR RESIDENTIAL COOLING
C NEFSTO - NUMBER OF ENERGY FORMS USED FOR RESIDENTIAL STOVES
C NEFDY - NUMBER OF ENERGY FORMS USED FOR RESIDENTIAL DRYERS
C NEFSHT - NUMBER OF ENERGY FORMS USED FOR RESIDENTIAL SECONDARY HEATING
C NEFWHR - NUMBER OF ENERGY FORMS USED FOR RESIDENTIAL HEATING
C ELINDEX - INDEX FOR ELECTRICITY IN COMMERCIAL DEMAND SUBMODULE
C ELHEATX - INDEX FOR ELECTRICITY IN RESIDENTIAL HEATING DEMAND TABLE
C ELWHEAX - INDEX FOR ELECTRICITY IN RESIDENTIAL WATER HEATING DEMAND TABLE
C ELCOOLX - INDEX FOR ELECTRICITY IN RESIDENTIAL COOLING DEMAND TABLE
C ELSTOVX - INDEX FOR ELECTRICITY IN RESIDENTIAL STOVES DEMAND TABLE
C ELDRYEX - INDEX FOR ELECTRICITY IN RESIDENTIAL DRYERS DEMAND TABLE
C ELSECHX - INDEX FOR ELECTRICITY IN RESIDENTIAL SECONDARY HEATING DEMAND TABLE
C ELLDVHX - INDEX FOR ELECTRICITY IN TRANSPORTATION, LIGHT DUTY ELECTR. VEHICLES
C ELRAILX - INDEX FOR ELECTRICITY IN TRANSPORTATION, RAILWAYS
C NEFELDV - NUMBER OF ENERGY FORM USED FOR ELECTRIC LIGHT DUTY VEHICLES
C NEFRAIL - NUMBER OF ENERGY FORM USED FOR TRAINS
C INDILDV - FIRST INDEX IN ARRAY FCLDV(INDILDV,NYRTR) ON COMMON /TCONSS/ TRANS.
C MAXECPs - MAXIMUM NUMBER OF SEGMENTS IN ECP LDC DEFINITION
C MAXITV - MAXIMUM NUMBER OF HOURLY INTERVALS PER SEGMENT IN ECP/efd LDC DEF.
C MAXECPB - MAXIMUM NUMBER OF BLOCKS IN ONE SEGMENT OF ECP LDC DEF.
C MAXECTB - MAXIMUM TOTAL NUMBER OF BLOCKS IN ECP LDC = MAXECPs*MAXECPB
C MAXEFDS - MAXIMUM NUMBER OF SEASONS USED BY EFD MODULE
C MAXEFDB - MAXIMUM NUMBER OF BLOCKS IN ONE SEASON LDC FOR EFD

C DISCFA - DISCOUNT FACTOR USED FOR COMPUTATIONS OF COSTS OF DSM OPTIONS
 C UNCONFA - CONVERSION FACTOR FOR UNITS BETWEEN INPUT AND OUTPUT DEMAND DATA
 C UNCONFAR - CONVERSION FACTOR FOR UNITS BETWEEN RESIDENTIAL INPUT DATA AND OUT.
 C MAXBLOCK - MAXIMUM NUMBER OF BLOCKS IN EFD LDC (MAXEFDS*MAXEFDB)
 C MAXRBT - MAXIMUM NUMBER OF RESIDENTIAL BUILDING TYPES USED IN LDSM
 C MAXCBT - MAXIMUM NUMBER OF COMMERCIAL BUILDING TYPES USED IN LDSM
 C MAXRDT - MAXIMUM NUMBER OF DECISION TYPES FOR RESIDENTIAL SECTOR
 C USED IN LDSM
 C MAXCOP - MAXIMUM NUMBER OF DSM OPTIONS CHOSEN BY ECP FROM COMMERCIAL SECTOR
 C MAXDSMP - MAXIMUM NUMBER OF DSM PROGRAMS FOR ONE REGION AND ONE SECTOR
 C MAXOPR - MAXIMUM NUMBER OF DSM OPTIONS PER DSM PROGRAM
 C MAXTAF - MAXIMUM NUMBER OF TECHNOLOGIES AFFECTED BY ONE DSM PROGRAM
 C MAXRDSMO - MAXIMUM NUMBER OF RESIDENTIAL DSM OPTIONS IN OPTION DATA BASE
 C MAXCDSMO - MAXIMUM NUMBER OF COMMERCIAL DSM OPTIONS IN OPTION DATA BASE
 C MAXNLST - MAXIMUM NUMBER OF LISTS OF REGIONS TO DEFINE DSM OPTIONS
 C MAXRLST - MAXIMUM NUMBER OF REGIONS+1 ON EACH OF THE ABOVE LISTS
 C MAXFRT - MAXIMUM NUMBER OF FROM TECHNOLOGIES AFFECTED BY ONE DSM OPTION
 C MAXTOT - MAXIMUM NUMBER OF TO TECHNOLOGIES AFFECTED BY ONE DSM OPTION
 C MAXRTECH - MAXIMUM NUMBER OF RESIDENTIAL TECHNOLOGIES TO BE AFFECTED BY DSM
 C MAXCTECH - MAXIMUM NUMBER OF COMMERCIAL TECHNOLOGIES TO BE AFFECTED BY DSM
 C MAXDMT - MAXIMUM NUMBER OF DEMAND MODULE TECHNOLOGIES AFFECTED BY ONE
 C DSM OPTION
 C MAXDECT - MAXIMUM NUMBER OF DECISION TYPES USED IN DSM OPTIONS
 C MAXRRST - MAXIMUM NUMBER OF RECORDS ON RESTART FILE FOR ONE SECTOR
 C NYRESTR - NUMBER OF YEARS ON RESIDENTIAL RESTART FILE
 C NYRESTC - NUMBER OF YEARS ON COMMERCIAL RESTART FILE
 C MAXNTPO - MAXIMUM NUMBER OF DEMAND MODULE TECHNOLOGIES PER OPTION (TO+FROM)
 C NUMCTCE - NUMBER OF ELEMENTS IN COMMERCIAL TECHNOLOGY CODE
 C TNSEFD - MAXIMUM TOTAL NUMBER OF SEGMENTS IN ALL SEASONS TOGETHER IN EFD LDC
 C MSEGFD - MAXIMUM NUMBER OF SEGMENTS PER SEASON IN EFD LDC
 C MAXREU - MAXIMUM NUMBER OF RESIDENTIAL SECTOR END-USES
 C MAXPCPH - MAXIMUM NUMBER OF COINCIDENT PEAK HOURS USED FOR "PCP PURPOSES"
 INTEGER*2 MAXPCPH
 INTEGER*2 MAXREU
 INTEGER*2 MSEGFD
 INTEGER*2 TNSEFD
 INTEGER*2 NUMCTCE
 INTEGER*2 MAXNTPO
 INTEGER*2 NYRESTR
 INTEGER*2 NYRESTC
 INTEGER*2 MAXRRST
 INTEGER*2 MAXDECT
 INTEGER*2 MAXDMT
 INTEGER*2 MAXRTECH,MAXCTECH
 INTEGER*2 MAXFRT,MAXTOT
 REAL*4 DISCFA
 INTEGER*2 MAXNLST
 INTEGER*2 MAXRLST
 INTEGER*2 MAXRDSMO,MAXCDSMO
 INTEGER*2 MAXTAF
 INTEGER*2 MAXDSMP
 INTEGER*2 MAXOPR
 REAL*4 UNCONFA
 REAL*4 UNCONFAR
 INTEGER*2 MAXCOP
 INTEGER*2 ELINDEX,ELHEATX,ELWHEAX,ELCOOLX,ELSTOVX,ELDRYEX,ELSECHX
 INTEGER*2 NYRRD,NCRRD,NEFHTR,NEFWHR,NEFCOO,NEFSTO,NEFDRY,NEFSHT
 INTEGER*2 NYRTR,NCRTR,NEFELDV,NEFRAIL,IND1LDV
 INTEGER*2 MAXECPB,MAXECP,MAXECTB
 INTEGER*2 MAXITV
 INTEGER*2 MAXEU
 INTEGER*2 MAXEFDS,MAXEFDB

```

INTEGER*2 MAXBLOCK
INTEGER*2 MAXSEC,RES,COM,IND,TRA
INTEGER*2 MAXREC
INTEGER*2 MAXMON,MAXSEA,MAXDTP
INTEGER*2 MAXNRG
INTEGER*2 MAXHOUR
INTEGER*2 MAXR HOUR
INTEGER*2 MAXDAY
INTEGER*2 MAXCRG
INTEGER*2 ELLDVHX
INTEGER*2 ELRAILX
INTEGER*2 MAXRBT,MAXRDT
INTEGER*2 MAXCBT
PARAMETER(MAXBLOCK=300)
PARAMETER(MAXEU=120) ! TARGET MAXEU=120
C* To add more sectors modify also DATA statement in subroutine DSMRDSTR
PARAMETER(MAXSEC=4,RES=1,COM=2,IND=3,TRA=4)
PARAMETER(MAXCRG=9)
PARAMETER(MAXREC=1600)
PARAMETER(MAXNRG=13)
PARAMETER(MAXMON=12,MAXDAY=3,MAXSEA=12,MAXDTP=3)
PARAMETER(MAXHOUR=MAXMON*MAXDAY*24+1)
PARAMETER(MAXR HOUR=8784)
PARAMETER(NYRRD=26,NCRRD=9,NEFHTR=7,NEFWHR=4,NEFCOO=3)
PARAMETER(NEFSTO=3,NEFDRIY=2,NEFSHT=7)
PARAMETER(ELINDEX=1,ELHEATX=2,ELWHEAX=2,ELCOOLX=1)
PARAMETER(ELSTOVX=3,ELDRYEX=2,ELSECHX=2)
PARAMETER(MAXECPB=3,MAXECP=9,MAXEFDS=6,MAXEFDB=50)
PARAMETER(MAXECTB=MAXECPB*MAXECP)
PARAMETER(MAXITV=2)
PARAMETER(ELLDVHX=6,NEFELDV=9,NYRTR=26,NCRTR=9,IND1LDV=16)
PARAMETER(ELRAILX=1,NEFRAIL=3)
PARAMETER(UNCONFA=292.83) ! currently: 1TBtu = 292.83GWh
PARAMETER(UNCONFAR=292.83E-06) ! currently: 1MBtu = 292.83E-06 GWh
PARAMETER(MAXRBT=3,MAXRDT=2)
PARAMETER(MAXCBT=11)
PARAMETER(MAXCOP=1500)
PARAMETER(MAXDSMP=12)
PARAMETER(MAXOPR=100)
PARAMETER(MAXTAF=100)
PARAMETER(MAXRDSMO=88,MAXCDSMO=88)
PARAMETER(MAXNLST=3,MAXRLST=13+1)
PARAMETER(DISCFA=0.05)
PARAMETER(MAXFRT=2,MAXTOT=2)
PARAMETER(MAXRTECH=37,MAXCTECH=10)
PARAMETER(MAXDMT=8)
PARAMETER(MAXDECT=3)
PARAMETER(MAXRRST=4520)
PARAMETER(NYRESTR=25)
PARAMETER(NYRESTC=26)
PARAMETER(MAXNTPO=8)
PARAMETER(NUMCTCE=3)
PARAMETER(TNSEEFD=18)
PARAMETER(MSEGEFD=3)
PARAMETER(MAXREU=10)
PARAMETER(MAXPCPH=20)
C===== (DSMEPRC) =====
C This INCLUDE file supplies definition of EPRICE variable
C=====
REAL*4 EPRICE(5,4,13) !indices: sector,function of system,NERC region
COMMON /CA/ EPRICE
C===== (DSMEUECS) =====

```

```

C This INCLUDE file supplies definition of RESIDENTIAL UEC variables
C=====
REAL*4 HTUEC,CLUUEC,HWUEC,CKUEC,DRYUEC,RFUEC,FZUEC,LTUEC,EAUEC
REAL*4 SHTUEC,APPUEC
COMMON/EUECS/HTUEC(9,11,3), CLUEC(9,5,3), HWUEC(9,4,3),
1CKUEC(9,3,3),DRYUEC(9,2,3),RFUEC(9,3),FZUEC(9,3),LTUEC(9,3),
2EAUEC(9,3),SHTUEC(9,7,3),APPUEC(9,3,3)
C===== DSMFMGRD =====
C*** Declarations of variables used with FILE_MGR
LOGICAL NEW ! File new or old?
CHARACTER*40 fname ! filename for calling FILE_MGR
EXTERNAL FILE_MGR
INTEGER*4 FILE_MGR ! FILE_MGR name declaration
C===== (DSMHHELM) =====
C**** This variables are used for communication within DSMHELM
INTEGER*2 K1 ! CURRENT YEAR NUMBER
INTEGER*2 RNB ! CURRENT REGION NUMBER
REAL*8 SYLOAD(MAXHOUR) ! system hourly loads
REAL*8 SectorLoad(MAXHOUR,MAXSEC) ! sectoral loads
REAL*8 SystemPeak(MNUMYR) ! System peaks for all years of ECP time horiz.
INTEGER*2 SystemPeakMonth(MNUMYR) !System peak month
INTEGER*2 SystemPeakDay(MNUMYR) !System peak day
INTEGER*2 SystemPeakHour(MNUMYR) !System peak hour
REAL*8 SystemLoadFactor(MNUMYR) !System load factor
REAL*8 TotSystemLoad(MNUMYR) !Total annual system load
COMMON/DSMHHLr/SYLOAD,SectorLoad,SystemPeak,SystemLoadFactor
&,TotSystemLoad
COMMON/DSMHHLi/K1,RNB,SystemPeakMonth,SystemPeakDay,SystemPeakHour
C===== (DSMNEMSC) =====
C USE INCLUDE's (DSMDIMEN) and (PARAMETR) with this file
C This COMMON BLOCK IS DESIGNED TO PASS RESULTS FROM LDSM TO THE REST OF NEMS
C ECPnumBl - Number of blocks in entire ECP LDC
C of ECP LDC
C ECPblockx(MAXECPs,MAXECPb) - x coordinates defining blocks in ECP LDC when
C hours are sorted by segment and by load
C ECPblWidth(MAXECTB) - widths of the blocks
C SumSegWidth - sum of segment widths
C BlockHeight(MAXECTB) ! heights of the blocks during ECP LDC computations
C ECPblSeg(MAXECTB) - segment to which the block belongs to
C ECPsgNum(MAXHOUR) - ECP LDC segment number for each hour of calendar year
C ECPsgFh(MAXECPs) ! first hour of segment
C ECPsgLh(MAXECPs) ! last hour of segment
C EFDsgNum(MAXHOUR) - Segment indices for all calendar hours
C PREVIOUSYR - previous run year !DEBUG~ONLY
C PREVIOUSITR - previous run iteration !DEBUG~ONLY
C PREVIOUSMODE - previous run mode of LDSM !DEBUG~ONLY
C ECPLastYearIndex - index of last year of time horizon for ECP
C NhourInRealSeason(MAXEFDS) - number of real hours in each EFD season
C NhourInRealSegment(TNSEEFD) - number of real hours in each EFD LDC segment
C numSEFD - total number of segments in the entire EFD LDC
C nLastCalHinSeg(TNSEEFD) - last calendar hour in each segment of EFD LDC
C nFirstCalHinSeg(TNSEEFD) - first calendar hour in each segment of EFD LDC
C nLastHinSeg(TNSEEFD) - last real hour in each segment of EFD LDC
C nFirstHinSeg(TNSEEFD) - first calendar hour in each segment of EFD LDC
C nLastBlockInSeason(MAXEFDS) - last block in each season of EFD LDC
C nFirstBlockInSeason(MAXEFDS) - first block in each season of EFD LDC
C NumberOfBlocksInSeg(TNSEEFD) - number of blocks in each segment of EFD LDC
C hCoordinate(MAXBLOCK) - coordinate on time axis during EFD LDC computations
REAL*4 hCoordinate(MAXBLOCK)
INTEGER*2 NumberOfBlocksInSeg(TNSEEFD)
INTEGER*2 nLastCalHinSeg(TNSEEFD)
INTEGER*2 nFirstCalHinSeg(TNSEEFD)

```

```

INTEGER*2 nLastHinSeg(TNSEEFD)
INTEGER*2 nFirstHinSeg(TNSEEFD)
INTEGER*2 nLastBlockInSeason(MAXEFDS)
INTEGER*2 nFirstBlockInSeason(MAXEFDS)
INTEGER*2 numSEFD
INTEGER*2 NhourInRealSeason(MAXEFDS)
INTEGER*2 NhourInRealSegment(TNSEEFD)
REAL*4 BlockHeight(MAXECTB)
INTEGER*4 ECPLastYearIndex
INTEGER*2 PREVIOUSMODE
INTEGER*2 PREVIOUSYR
INTEGER*2 PREVIOUSITR
REAL*4 ECPblWidth(MAXECTB)
REAL*4 ECPblockx(MAXECPS,MAXECPB)
INTEGER*2 SumSegWidth
INTEGER*2 ECPblSeg(MAXECTB)
INTEGER*2 ECPsgNum(MAXHOUR)
INTEGER*2 ECPsgFh(MAXECPS)
INTEGER*2 ECPsgLh(MAXECPS)
INTEGER*2 ECPnumBl
INTEGER*2 EFDsgNum(MAXHOUR)
COMMON/DSMNMi/ECPLastYearIndex,ECPnumBl,ECPsgNum,SumSegWidth,
&ECPblSeg,ECPsgFh,ECPsgLh,PREVIOUSYR,PREVIOUSITR
&,EFDsgNum,numSEFD,NhourInRealSeason
&,NhourInRealSegment,nLastCalHinSeg,nFirstCalHinSeg
&,nLastHinSeg,nFirstHinSeg,nFirstBlockInSeason,nLastBlockInSeason
&,NumberOfBlocksInSeg
COMMON/DSMNMmr/ECPblWidth,ECPblockx,BlockHeight,hCoordinate
C ===== (DSMNERCR) ===== ! updated
C NERCnam(MAXNRG) - NERC REGION NAMES
C NERCtdloss(MAXNRG) - transmission and distribution loss factors for NERC reg.
C NERClsrN(MAXNRG) - DAF-LSR-DB record numbers for regional system load lsrs
C nNERCreg - number of NERC regions to be processed
C nCENSUSreg - number of CENSUS regions to be processed
CHARACTER*8 NERCnam(MAXNRG)
INTEGER*2 nNERCreg,nCENSUSreg,NERClsrN(MAXNRG)
REAL*4 NERCtdloss(MAXNRG)
COMMON/DSMNMrn/NERCtdloss,NERClsrN,nNERCreg,nCENSUSreg
COMMON/DSMNMrc/NERCnam
C===== (DSMOPTDB) =====
C This INCLUDE file is used for definition of the DSM Options Data Base
C=====
C RprogCode(MAXDSMP) - list of codes identifying Residential DSM programs
C CprogCode(MAXDSMP) - list of codes identifying Commercial DSM programs
C RtechNumb - number of residential technologies potentially affected by DSM
C CtechNumb - number of commercial technologies potentially affected by DSM
C RtechCode(MAXRTECH) - list of residential technology codes
C CtechCode(MAXRTECH,NUMCTCE) - list of commercial technology 3-element codes
C 1st element - SERVICE ,2nd - VINTAGE, 3rd - TECHNOLOGY
C RtechDMtn(MAXRTECH) - number of DEMAND MODULE technologies represented by
C RtechDMG(MAXRTECH,MAXDMT) - list of technology group indices
C RtechDMT(MAXRTECH,MAXDMT) - list of technology indices within techn. groups
C RtechLSR(MAXRTECH) - number of record on the DAF-LSR-DB with appropriate LSR
C CtechLSR(MAXRTECH,MAXCBT) - number of record on the DAF-LSR-DB with appr.LSR
C RtechRrc(MAXRTECH,MAXDMT,MAXNRG,MAXRBT) - Resid. RESTATR FILE record numbers
C CtechRrc(MAXRTECH,MAXNRG,MAXCBT) - Commercial RESTATC FILE record numbers
C RRListN - number of region groups use during residential DSM option definition
C CRListN - number of region groups use during commercial DSM option definition
C RRListID(MAXNLST) - list identifiers CHARACTER*1
C CRListID(MAXNLST) - list identifiers CHARACTER*1
C RdecTYPn - number of decision types in residential sector
C CdecTYPn - number of decision types in commercial sector

```

C RdecTYPid(MAXDECT) - decision type identifiers in residential sector
 C CdecTYPid(MAXDECT) - decision type identifiers in commercial sector
 C RdecTYPix(MAXDECT) - decision type indices in residential sector
 C CdecTYPix(MAXDECT) - decision type indices in commercial sector
 C RbuildTn - number of building types in residential sector
 C CbuildTn - number of building types in commercial sector
 C RbuildTid(MAXRBT) - building types identifiers in residential sector
 C CbuildTid(MAXCBT) - building types identifiers in commercial sector
 C RbuildTix(MAXRBT) - building types indices in residential sector
 C CbuildTix(MAXCBT) - building types indices in commercial sector
 C RrLIST(MAXNLST,MAXRLST) - lists of regions to which residential options apply
 C attention: first item on the list is the number of region indices on the list
 C CrLIST(MAXNLST,MAXRLST) - lists of regions to which commercial options apply
 C attention: first item on the list is the number of region indices on the list
 C DSMROptionNumb - Number of DSM options available for residential sector
 C DSMROptionCode(MAXRDSMO) - vector of residential sector DSMOption code names
 C DSMROptionRegion(MAXRDSMO) - # of list of regions to which the option applies
 C DSMROptionBuildT(MAXRDSMO) - # of building type to which the option applies
 C DSMROptionDecType(MAXRDSMO) - # of decision type to which the option applies
 C DSMROptionFromTnum(MAXRDSMO) - number of FROM technologies affected by option
 C DSMROptionFromTech(MAXRDSMO,MAXFRT) - indices of FROM technologies
 C DSMROptionToTnum(MAXRDSMO) - number of TO technologies affected by option
 C DSMROptionToTech(MAXRDSMO,MAXTOT) - indices of TO technologies
 C DSMROptionCost(MAXRDSMO) - Marketing/administrative costs of option /kWh saved
 C DSMROptionFyr(MAXRDSMO) - first year when the option is available (index)
 C DSMROptionRamp(MAXRDSMO) - number of ramp-up years for the option
 C DSMROptionRebate(MAXRDSMO,MAXNRG) - current rebate value for an option
 C NRPROG - number of residential DSM programs
 C RPROG(MAXRDSMO) - assignment of residential DSM options to programs
 C RRSTRN(MAXRTECH,MAXNRG) - RESTART file record numbers with data for the
 C technologies
 C
 C CrLIST(MAXNLST,MAXRLST) - lists of regions to which commercial options apply
 C DSMCOptionNumb - Number of DSM options available for commercial sector
 C DSMCOptionCode(MAXCDSMO) - vector of commercial sector DSM option code names
 C DSMCOptionRegion(MAXCDSMO) - # of list of regions for the option
 C DSMCOptionBuildT(MAXCDSMO) - building type to which the option applies
 C DSMCOptionDecType(MAXCDSMO) - decision type to which the option applies
 C DSMCOptionFromTnum(MAXCDSMO) - number of FROM technologies affected by option
 C DSMCOptionFromTech(MAXCDSMO,NUMCTCE) - FROM technology KTECH codes for option
 C 1-service #, 2 - vintage #, 3 - technology # ?
 C DSMCOptionToTnum(MAXCDSMO) - number of TO technologies affected by option
 C DSMCOptionToTech(MAXCDSMO,NUMCTCE) - TO technology KTECH codes for the option
 C 1-service #, 2 - vintage #, 3 - technology # ?
 C DSMCOptionCost(MAXCDSMO) - Marketing/administrative costs of the option
 C DSMCOptionFyr(MAXCDSMO) - first year when the option is available
 C DSMCOptionRamp(MAXCDSMO) - number of ramp-up years of the option
 C DSMCOptionRebate(MAXCDSMO,MAXNRG) - current rebates
 C NCPROG - number of commercial DSM programs
 C CPROG(MAXCDSMO) - assignment of commercial DSM options to programs
 C CRSTRN(MAXCTECH,MAXNRG) - RESTART file record numbers with data for the
 C technologies
 C PAYBACK - DSM option pay-back period
 INTEGER*2 RrLIST(MAXNLST,MAXRLST)
 INTEGER*2 CrLIST(MAXNLST,MAXRLST)
 INTEGER*2 DSMROptionFromTnum(MAXRDSMO)
 INTEGER*2 DSMCOptionFromTnum(MAXCDSMO)
 INTEGER*2 DSMCOptionToTnum(MAXCDSMO)
 INTEGER*2 DSMROptionToTnum(MAXRDSMO)
 INTEGER*2 PAYBACK
 INTEGER*4 RRSTRN(MAXRTECH,MAXNRG),CRSTRN(MAXCTECH,MAXNRG)
 INTEGER*2 NRPROG,RPROG(MAXRDSMO),NCPROG,CPROG(MAXCDSMO)

INTEGER*2 DSMROptionNumb
 CHARACTER*6 DSMROptionCode(MAXRDSMO)
 INTEGER*2 DSMROptionRegion(MAXRDSMO)
 INTEGER*2 DSMROptionBuildT(MAXRDSMO)
 INTEGER*2 DSMROptionDecType(MAXRDSMO)
 INTEGER*2 DSMROptionFromTech(MAXRDSMO,MAXFRT)
 INTEGER*2 DSMROptionToTech(MAXRDSMO,MAXTOT)
 REAL*4 DSMROptionCost(MAXRDSMO)
 INTEGER*2 DSMROptionFyr(MAXRDSMO)
 INTEGER*2 DSMROptionRamp(MAXRDSMO)
 INTEGER*2 DSMCOptionNumb
 CHARACTER*6 DSMCOptionCode(MAXCDSMO)
 INTEGER*2 DSMCOptionRegion(MAXCDSMO)
 INTEGER*2 DSMCOptionBuildT(MAXCDSMO)
 INTEGER*2 DSMCOptionDecType(MAXCDSMO)
 INTEGER*2 DSMCOptionFromTech(MAXCDSMO,NUMCTCE)
 INTEGER*2 DSMCOptionToTech(MAXCDSMO,NUMCTCE)
 REAL*4 DSMCOptionCost(MAXCDSMO)
 INTEGER*2 DSMCOptionFyr(MAXCDSMO)
 INTEGER*2 DSMCOptionRamp(MAXCDSMO)
 CHARACTER*7 RprogCode(MAXDSMP)
 CHARACTER*7 CprogCode(MAXDSMP)
 INTEGER*2 RtechNumb,CtechNumb
 CHARACTER*7 RtechCode(MAXRTECH)
 INTEGER*4 CtechCode(MAXCTECH,NUMCTCE)
 INTEGER*2 RtechDMtn(MAXRTECH)
 INTEGER*2 RtechDMG(MAXRTECH,MAXDMT)
 INTEGER*2 RtechDMT(MAXRTECH,MAXDMT)
 INTEGER*4 RtechLSR(MAXRTECH)
 INTEGER*4 CtechLSR(MAXCTECH,MAXCBT)
 INTEGER*4 RtechRrc(MAXRTECH,MAXDMT,MAXNRG,MAXRBT)
 INTEGER*4 CtechRrc(MAXCTECH,MAXNRG,MAXCBT)
 INTEGER*2 RRlistN,CrlistN
 CHARACTER*1 RRlistID(MAXNLIST),CrlistID(MAXNLIST)
 INTEGER*2 RdecTYPn,CdecTYPn
 CHARACTER*1 RdecTYPid(MAXDECT),CdecTYPid(MAXDECT)
 INTEGER*2 RdecTYPix(MAXDECT),CdecTYPix(MAXDECT)
 INTEGER*2 RbuildTn,CbuildTn
 CHARACTER*2 RbuildTid(MAXRBT),CbuildTid(MAXCBT)
 INTEGER*2 RbuildTix(MAXCBT),CbuildTix(MAXCBT)
 REAL*4 DSMROptionRebate(MAXRDSMO,MAXNRG)
 REAL*4 DSMCOptionRebate(MAXCDSMO,MAXNRG)
 COMMON /DSMOPi/RRSTRN,CRSTRN,CtechCode,RtechLSR,CtechLSR,
 & RtechRrc,CtechRrc,
 & DSMROptionNumb,DSMROptionRegion,DSMROptionBuildT,
 & DSMROptionDecType,DSMROptionFyr,DSMROptionRamp,DSMCOptionNumb,
 & DSMCOptionBuildT,DSMCOptionDecType,RrLIST,CrLIST,
 & DSMCOptionFyr,
 & DSMCOptionRamp,NRPROG,NCPROG,RPROG,CPROG,
 & RtechNumb,RtechDMtn,RtechDMG,RtechDMT,RRlistN,
 & CtechNumb,CrlistN,
 & RdecTYPn,RdecTYPix,RbuildTn,RbuildTix,PAYBACK,DSMROptionToTech,
 & CdecTYPn,CdecTYPix,CbuildTn,CbuildTix,DSMCOptionToTech,
 & DSMROptionFromTnum,DSMROptionToTnum,DSMROptionFromTech,
 & DSMCOptionFromTnum,DSMCOptionToTnum,DSMCOptionFromTech,
 & DSMCOptionRegion
 COMMON /DSMOPr/DSMROptionCost,DSMCOptionCost,DSMROptionRebate,
 & DSMCOptionRebate
 COMMON /DSMOPc/DSMROptionCode,
 & DSMCOptionCode,
 & RprogCode,CprogCode,RtechCode,RRlistID,RdecTYPid,RbuildTid
 & ,CrlistID,CdecTYPid,CbuildTid

```

C ===== (DSMPROG) =====
C LSRindex(MAXTAF,MAXDSMP) - DAF-LSR-DB record numbers
C DeltaEnergy(MAXTAF,ECP$XPH,ECP$XPH,MAXDSMP) -PROGRAM'S IMPACT ON ANNUAL ENERG
C DEMAND BY TECHNOLOGY, PROGRAM START YEAR, PROGRAM YEAR, NERC REGION
C RPROGDEFN(MAXNRG,MAXDSMP) - number of options in each residential DSM program
C CPROGDEFN(MAXNRG,MAXDSMP) - number of options in each commercial DSM program
C RPROGDEF(MAXNRG,MAXDSMP,MAXOPR,2) - 1, indices of options in resid. DSM pr.
C
C 2, status of options
C CPROGDEF(MAXNRG,MAXDSMP,MAXOPR,2) - 1, indices of options in comm. DSM prog.
C
C 2, status of options
C NTECHAFFP(MAXDSMP) - number of technologies affected by a DSM program
  INTEGER*2 RPROGDEFN(MAXNRG,MAXDSMP),CPROGDEFN(MAXNRG,MAXDSMP)
  & ,RPROGDEF(MAXNRG,MAXDSMP,MAXOPR,2)
  & ,CPROGDEF(MAXNRG,MAXDSMP,MAXOPR,2)
  REAL*4 DeltaEnergy(MAXTAF,ECP$XPH,ECP$XPH,MAXDSMP)
  INTEGER*2 LSRindex(MAXTAF,MAXDSMP)
  INTEGER*2 NTECHAFFP(MAXDSMP)
  COMMON /DSMDPr/ DeltaEnergy
  COMMON /DSMDPi/ RPROGDEFN,CPROGDEFN,RPROGDEF,CPROGDEF
  & ,LSRindex,NTECHAFFP
C ===== (DSMRETDDB) =====
C This INCLUDE file specifies variables with RESIDENTIAL technology data
C =====
  INTEGER*2 EquipLife(5,9) ! temporary variable with equipment lifetime
  REAL*4 Cost(5,9) ! temporary variable with equipment costs
  COMMON/DSMRETN/ Cost,EquipLife
C ===== (DSMSECTR) ===== ! updated
C SEC(MAXSEC) - SECTOR INDICES IN ORDER
C SecNam(MAXSEC) - ABBREVIATED SECTOR NAMES
C NumSec - NUMBER OF SECTORS
C SLNAM(MAXSEC) - FULL LENGTH SECTOR NAMES
C NEUSES(MAXSEC) - NUMBER OF END-USES IN SECTORS
C EUINDEX(MAXSEC,2) - FIRST/LAST END-USE NUMBER FOR A GIVEN SECTOR
C EUNAM(MAXEU) - END-USE NAMES
C LoadForec(MAXEU,2) - TRADITIONAL/DELTA approach load forecast by end-use
C EURECNUM(MAXEU,MAXNRG) - DAF-LSR-DB record number for lsr for an eu
C Neu - Total number of end-uses defined on the structure file
C LFinum - number of items on LoadForec array
C BaseYrLd(MAXEU,MAXNRG) - Base year load by end-use and region
C BaseYrSysLd(MAXNRG) - Base year system load by NERC region
C SystemLoad - Total system load for a region
C TotDemCensus - Total Demand in Census regions
C TotDemNERC - Total Demand in NERC regions
C ResTGdem(MAXREU) - Current region, year demand for end-uses in residential s.
C ComDemAdjFac(MNUMCR) - Adjustment factors for Commercial load by census div.
C MappCtoN(MNUMNR,MNUMCR,MAXSEC) - matrix used for mapping CENSUS to NERC reg.
C tells what fraction of a CENSUS value contributes to a NERC value
C MappNtoC(MNUMNR,MNUMCR,MAXSEC) - matrix used for mapping NERC to CENSUS div.
C tells what fraction of a NERC value contributes to a CENSUS value
  REAL*4 MappCtoN(MNUMNR,MNUMCR,MAXSEC)
  REAL*4 MappNtoC(MNUMNR,MNUMCR,MAXSEC)
  REAL*4 ComDemAdjFac(MNUMCR)
  REAL*4 ResTGdem(MAXREU)
  REAL*4 TotDemCensus,TotDemNERC
  CHARACTER*3 SecNam(MAXSEC)
  CHARACTER*12 SLNAM(MAXSEC)
  CHARACTER*40 EUNAM(MAXEU)
  INTEGER*2 EUINDEX(MAXEU,2)
  INTEGER*2 NumSec
  INTEGER*2 EURECNUM(MAXEU,MAXNRG)
  INTEGER*2 SEC(MAXSEC),NEUSES(MAXSEC)
  INTEGER*2 Neu

```

```

INTEGER*2 LFinum
REAL*4 LoadForec(MAXEU,2)
REAL*4 BaseYrLd(MAXEU,MAXNRG)
REAL*4 BaseYrSysLd(MAXNRG)
REAL*4 SystemLoad
COMMON/DSMSCn/LFinum,NumSec,SEC,NEUSES,EUINDEX,
&EUrecNUM,Neu
COMMON/DSMSCr/LoadForec,SystemLoad,BaseYrLd,
&BaseYrSysLd,TotDemCensus,TotDemNERC,ResTGdem,ComDemAdjFac,
&MappCtoN
COMMON/DSMSCc/SecNam,SLNAM,EUNAM
C===== (DSMTFCOM) =====
C This INCLUDE file is used for communication with Commercial Demand Module
C=====
C USE INCLUDE's (DSMDIMEN) and (PARAMETR) with this file
C DSMcOptionsNumber(MAXCRG) - number of DSM options chosen for each of census
C     regions
C DSMcOptionIndex(MAXCOP) - list of indices of DSM options chosen by ECP
C     (indices as on DSM Option Data Base which is
C     loaded into a COMMON block at the beginning of NEMS
C     run; "FROM" and "TO" technologies on the data base
C     are depicted by identifiers from KTECH data base,
C     each option applies to a specified
C     building type and decision type)
C DSMcFracOptionMarket(MAXCOP) - fraction of the market available for an
C     option from the list, decided to be subjected
C     to the option
C DSMcServDemChange(MAXCOP) - actual change in service demand resulting from
C     application of DSM options, calculated by the
C     Commercial Demand Forecasting Module
C     (same order as on DSMcFracOptionMarket)
C MAXCRG - constant defined on DSMDIMEN, maximum of census regions = MNUMCR-2
C DSM options database is defined in the INCLUDE file DSMOPTDB
C=====
INTEGER*2 DSMcOptionsNumber(MAXCRG)
INTEGER*2 DSMcOptionIndex(MAXCOP)
REAL*4 DSMcFracOptionMarket(MAXCOP)
REAL*4 DSMcServDemChange(MAXCOP)
COMMON /DSMTFCM/DSMcServDemChange,DSMcFracOptionMarket,
& DSMcOptionsNumber,DSMcOptionIndex
C===== (DSMTFECP) =====
C This INCLUDE file is used for communication with the ECP Module
C=====
C USE INCLUDE's (DSMDIMEN) and (PARAMETR) with this file
C DSMNPROG - number of DSM programs to be considered
C DSMPRCST(ECP$XPH,MAXNRG,MAXDSMP) - DSM program costs (objective function co.)
C     dimensions: program start year,region,program
C DSMPRLIM(ECP$XPH,ECP$XPH,MAXNRG,MAXECTB,MAXDSMP) - DSM program load impact
C     dimensions: program start year, program year, region, LDC block, program #
C ECPLDCBH(ECP$XPH,MAXNRG,MAXECTB) - ECP LDC block heights
C ECPLDCBS(ECP$XPH,MAXNRG,MAXECTB) - ECP LDC block segment assignment
C ECPLDCBW(ECP$XPH,MAXNRG,MAXECTB) - ECP LDC block widths
C ECPnumSg - Number of segments in LDC for ECP module
C ECPsgDblock(MAXECPs,MAXECPb) - %% of hours allocated to blocks in each segment
C ECPsgDbltyp(MAXECPs,MAXECPb) - type peak/non-peak of a block
C ECPsgDnB(MAXECPs) - Number of blocks in each of segments of ECP LDC
C DSMPRCHOICE(MAXNRG,MAXDSMP) - choice of DSM programs made by ECP
C RopChoice(MNUMYR,MAXNRG,MAXRDSMO) - choice of resid DSM options for next year
C CopChoice(MNUMYR,MAXNRG,MAXCDSMO) - choice of comm. DSM options for next year
C BlockNum(MAXECTB) - order in which blocks are sorted in ECP LDC
C=====
REAL*4 RopChoice(MNUMYR,MAXNRG,MAXRDSMO)

```



```

REAL*4 CopChoice(MNUMYR,MAXNRG,MAXCDSMO)
INTEGER*2 BlockNum(MAXECTB)
REAL*4 DSMPRCHOICE(MAXNRG,MAXDSMP)
INTEGER*2 DSMNPROG
CHARACTER*1 ECPsgDblytyp(MAXECPS,MAXECPB)
INTEGER*2 ECPLDCBS(MNUMYR,MAXNRG,MAXECTB)
INTEGER*2 ECPnumSg
INTEGER*2 ECPsgDnB(MAXECPS)
REAL*4 DSMPRCST(ECP$XPH,MAXNRG,MAXDSMP)
REAL*4 DSMPRLIM(ECP$XPH,ECP$XPH,MAXNRG,MAXECTB,MAXDSMP)
REAL*4 ECPLDCBH(MNUMYR,MAXNRG,MAXECTB)
REAL*4 ECPLDCBW(MNUMYR,MAXNRG,MAXECTB)
REAL*4 ECPsgDblock(MAXECPS,MAXECPB)
COMMON /DSMECt/DSMPRCST,DSMPRLIM,ECPLDCBH,ECPLDCBW,ECPsgDblock,
&DSMPRCHOICE,RopChoice,CopChoice
COMMON /DSMECt/ECPLDCBS,ECPnumSg,ECPsgDnB,DSMNPROG,BlockNum
COMMON /DSMECc/ECPsgDblytyp

```

```

C===== (DSMTFEFP) =====

```

```

C This INCLUDE file is used for communication with the EFP module

```

```

C=====

```

```

C SecLoad(MNUMNR,MAXSEC) - Total sectorial load for current year, by NERC reg
C SecMonLdPeak(MNUMNR,MAXSEC,MAXMON,2) - Coincident/Noncoincident sectorial
C monthly peaks for a current year by NERC regions
C SecAnnulPeak(MNUMNR,MAXSEC,2) - Coincident/Noncoincident sectorial annual peak
C for current year by NERC regions
C SecAnnPeaAvPCP(MNUMNR,MAXSEC) - Averages of sectorial loads from top NpeakH
C system peak (real) hours (coincident peaks) for "PCP purposes"
C SystemLF(MNUMNR) - system load factor for CURIYR and each NERC region
C NpeakH - number of hours used for calculation of SecAnnPeakAvPCP values
C SysPeakHour(MAXPCPH) - NpeakH system peak load hours in descending order
C DSMAnnualCost(MNUMNR,MAXSEC) - Annual DSM costs by sector
REAL*4 DSMAnnualCost(MNUMNR,MAXSEC)
REAL*4 TotSecLoad(MNUMNR,MAXSEC),SystemLF(MNUMNR)
REAL*4 SecMonLdPeak(MNUMNR,MAXSEC,MAXMON,2)
REAL*4 SecAnnualPeak(MNUMNR,MAXSEC,2)
REAL*4 SecAnnPeaAvPCP(MNUMNR,MAXSEC)
INTEGER*2 SysPeakHour(MAXPCPH)
INTEGER*2 NpeakH
COMMON/DSMEFPn/TotSecLoad,SecMonLdPeak,SecAnnualPeak,SystemLF,
& SecAnnPeaAvPCP,DSMAnnualCost,SysPeakHour,NpeakH

```

```

C===== (DSMTFRES) =====

```

```

C This INCLUDE file is used for communication with Residential Demand Module

```

```

C=====

```

```

C USE INCLUDE's (DSMDIMEN) and (PARAMETR) with this file
C DSMrOptionsNumber(MAXCRG) - number of DSM options chosen for each of census
C regions
C DSMrOptionIndex(MAXCOP) - list of indices of DSM options chosen by ECP
C (indices as on DSM Option Data Base which is
C loaded into a COMMON block at the beginning of NEMS
C run; "FROM" and "TO" technologies on the data base
C are depicted by identifiers from KTECH data base,
C each option applies to a specified
C building type and decision type)
C DSMrFracOptionMarket(MAXCOP) - fraction of the market available for an
C option from the list, decided to be subjected
C to the option
C DSMrNumUnitChange(MAXCOP) - actual change in number of units resulting from
C application of DSM options, calculated by the
C Residential Demand Forecasting Module
C (same order as on DSMrFracOptionMarket)
C MAXCRG - constant defined on DSMDIMEN, maximum of census regions = MNUMCR-2
C DSM options database is defined in the INCLUDE file DSMOPTDB

```

```

C=====
  INTEGER*2 DSMrOptionsNumber(MAXCRG)
  INTEGER*2 DSMrOptionIndex(MAXCOP)
  REAL*4 DSMrFracOptionMarket(MAXCOP)
  REAL*4 DSMrNumUnitChange(MAXCOP)
  COMMON /DSMTFRS/DSMrNumUnitChange,DSMrFracOptionMarket,
&      DSMrOptionsNumber,DSMrOptionIndex
C===== (DSMTOEFD) =====
C   This INCLUDE file is used for communication with the EFD module
C=====
C USE INCLUDE's (DSMDIMEN) and (PARAMETR) with this file
C EFDLDCYC(MAXNRG,MAXEFDS,MAXEFDP) - Y coordinates of EFD LDC data points
C EFDLDCPR(MAXNRG,MAXEFDS,MAXEFDP) - EFD LDC data point ranks in segments
C EFDLDCSA(MAXNRG,MAXEFDS,MAXEFDP) - EFD LDC data point segment assignment
C EFDLDCBW(MAXEFDS) - EFD LDC block widths
C EFDnS - EFD LDC number of seasons
C EFDnumBI(MAXEFDS) - Number of EFD LDC blocks for each season
C EFDnumSeg(MAXEFDS) - Number of segments in LDC's for EFD module
C=====
  INTEGER*2 EFDLDCPR(MAXNRG,MAXEFDS,MAXEFDB)
  INTEGER*2 EFDLDCSA(MAXNRG,MAXEFDS,MAXEFDB)
  INTEGER*2 EFDnS
  INTEGER*2 EFDnumBI(MAXEFDS)
  INTEGER*2 EFDnumSeg(MAXEFDS)
  REAL*4 EFDLDCYC(MAXNRG,MAXEFDS,MAXEFDB)
  REAL*4 EFDLDCBW(MAXEFDS)
  COMMON /DSMFDi/EFDLDCPR,EFDLDCSA,EFDnS,EFDnumBI,EFDnumSeg
  COMMON /DSMFDr/EFDLDCYC,EFDLDCBW
C===== (DSMUNITS) =====
C IMSG - LDSM MESSAGE FILE
C IODB - DAF-LSR-DB FILE
C IORR - RESIDENTIAL RESTART FILE
C IOCR - COMMERCIAL RESTART FILE
  INTEGER*4 IMSG
  INTEGER*4 IODB,IORR,IOCR
  COMMON/DSMUNn/IMSG,IODB,IORR,IOCR

```

Cross Reference Table of Source Code Variables

#	Variable name	Numbers of lines of the code that include the variable
1	LDSMODE	1, 12, 30, 56, 86
2	WHOOPS	15, 26, 44, 45, 74, 75, 87, 88, 104, 119, 157, 160, 175, 212, 215, 234, 360, 365, 368, 383, 423, 442, 598, 611, 631, 632, 998, 1016, 1030, 1031, 1033, 1034, 1036, 1037, 1039, 1040, 1071, 1354, 1418, 1828, 1833, 1839, 1845, 2649, 2714, 2758, 2759, 2870, 2871, 2877, 2878, 2890, 2891, 3029, 3030, 3043, 3044, 3046, 3047, 3052, 3053, 3065, 3066, 3074, 3075, 3094, 3095, 3123, 3124, 3126, 3127, 3130, 3131, 3133, 3134, 3137, 3138, 3141, 3142, 3144, 3145, 3148, 3149, 3153, 3154, 3168, 3169, 3171, 3172, 3174, 3175, 3188, 3189, 3192, 3193, 3197, 3198, 3201, 3202, 3207, 3208, 3210, 3211, 3297, 3298, 3303, 3304, 3319, 3320, 3322, 3323, 3325, 3326, 3339, 3340, 3343, 3344, 3348, 3349, 3352, 3353, 3358, 3359, 3361, 3362, 3473, 3478, 3482, 3486, 3491, 3496, 3501, 3506, 3511, 3516, 3521, 3526, 3531, 3536, 3541, 3546, 3551, 3554, 3569, 3582
3	TMPRNB	16, 71, 72
4	TOTDEMNERC	29
5	CURIYR	30, 58, 66, 70, 72, 135, 141, 190, 195, 261, 262, 267, 276, 285, 294, 303, 312, 321, 330, 339, 348, 399, 404, 412, 455, 459, 618, 620, 628, 629, 673, 676, 679, 690, 698, 709, 718, 732, 750, 1105, 1344, 1453, 1499, 1500, 1511, 1540, 1541, 1552, 1572, 1576, 1603, 1604, 1608, 1643, 1684, 1685, 1696, 1724, 1725, 1736, 1755, 1758, 1786, 1787, 1791, 1886, 1906, 1958, 2592, 2605, 3606, 3607, 3608, 3609, 3610, 3611
6	FIRSYR	30, 58, 70, 628, 689, 708, 1046, 1577, 1759
7	CURITR	30, 70, 86, 1046
8	NEW	31, 33, 34, 36, 37, 39, 41, 42, 2755, 2757, 3467
9	FNAME	32, 33, 35, 36, 38, 39, 40, 41, 2756, 2757, 3467
10	IMSG	33
11	FILE_MGR	33, 36, 39, 41, 2757, 3467
12	IODB	36, 1331, 1969, 2746, 2752, 3274, 3748, 3790, 3969, 3990
13	IOCR	39, 1675, 1713, 3447, 3449
14	IORR	41, 1490, 1531, 3275, 3277
15	RNB	61, 71, 145, 198, 270, 279, 288, 297, 306, 315, 324, 333, 342, 351, 407, 415, 592, 594, 858, 859, 860, 863, 864, 865, 866, 867, 868, 898, 905, 912, 914, 1050, 1054, 1059, 1061, 1112, 1121, 1133, 1135, 1139, 1263, 1264, 1265, 1349, 1457, 1471, 1473, 1485, 1494, 1526, 1534, 1563, 1580, 1632, 1647, 1659, 1661, 1671, 1678, 1711, 1746, 1762, 1815, 1874, 1875, 1876, 1877, 1889, 1894, 1899, 1909, 1914, 1919, 1981, 1987, 2286, 2290, 2320, 2324, 2416, 2421, 2426, 2431, 2436, 2441, 2446, 2451, 2456, 2461, 2482, 2487, 2492, 2497, 2502, 2507, 2512, 2517, 2522, 2527, 3606, 3607, 3608, 3609, 3610, 3611, 3719, 3723, 3761, 3765, 3856, 4054
16	NNERCREG	61, 556, 683, 731, 749, 945, 969, 988, 2735, 3031, 3076, 3104, 3111, 3178, 3281, 3329, 3452
17	K1	66, 135, 190, 262, 399, 620, 629, 898, 914, 1046, 1105, 1147, 1148, 1149, 1150, 1263, 1264, 1265, 1344, 1958, 3948
18	LOAD	117, 173, 228, 381, 440, 452, 455, 458, 459, 1014, 1057, 1059, 1061, 1938, 1971, 1973
19	L	118, 133, 147, 150, 151, 174, 188, 203, 205, 206, 229, 257, 274, 283, 292, 301, 310, 319, 328, 337, 346, 354, 382, 397, 410, 417, 441, 884, 929, 1088, 1180, 1221, 1227, 1230, 1231, 1311, 1387, 1484, 1485, 1486, 1525, 1526, 1527, 1621, 1622,

20

I

1623, 1625, 1626, 1627, 1669, 1670, 1706,
 1707, 1802, 1803, 1804, 1805, 1807, 1808,
 1809, 1810, 1936, 2233, 2280, 2281, 2314,
 2315, 2367, 2408, 2409, 2410, 2412, 2414,
 2417, 2419, 2422, 2424, 2427, 2432, 2437,
 2442, 2444, 2447, 2449, 2452, 2454, 2457,
 2459, 2474, 2475, 2476, 2478, 2480, 2483,
 2485, 2488, 2490, 2493, 2498, 2503, 2508,
 2510, 2513, 2515, 2518, 2520, 2523, 2525,
 2665, 2751, 2752, 2805, 2807, 2814, 2820,
 2896, 2904, 2905, 2908, 2914, 2915, 3033,
 3034, 3035, 3079, 3080, 3081, 3102, 3104,
 3106, 3107, 3108, 3111, 3114, 3116, 3117,
 3158, 3159, 3160, 3251, 3252, 3253, 3261,
 3262, 3263, 3276, 3277, 3282, 3283, 3287,
 3291, 3308, 3309, 3310, 3416, 3419, 3422,
 3430, 3433, 3436, 3448, 3449, 3453, 3454,
 3458, 3462, 3636, 3713, 3714, 3724, 3726,
 3746, 3750, 3755, 3756, 3766, 3768, 3788,
 3792, 3889, 3959, 3961, 3962, 3963, 3964,
 3980, 3981, 3982, 3983, 3984

118, 137, 141, 174, 193, 441, 453, 455,
 483, 496, 497, 499, 500, 503, 504, 506,
 507, 511, 512, 514, 515, 519, 520, 522,
 523, 573, 591, 592, 654, 672, 673, 675,
 676, 695, 696, 697, 698, 699, 714, 715,
 716, 717, 718, 719, 786, 809, 810, 820,
 842, 850, 884, 897, 898, 902, 905, 908,
 909, 912, 929, 945, 947, 950, 952, 959,
 960, 962, 966, 967, 978, 979, 981, 985,
 986, 1013, 1028, 1029, 1032, 1035, 1038,
 1049, 1050, 1053, 1054, 1058, 1059, 1088,
 1106, 1108, 1115, 1116, 1130, 1131, 1133,
 1135, 1141, 1142, 1143, 1144, 1152, 1153,
 1156, 1180, 1203, 1204, 1208, 1209, 1252,
 1253, 1262, 1263, 1264, 1265, 1387, 1465,
 1471, 1473, 1494, 1497, 1534, 1538, 1563,
 1623, 1627, 1632, 1638, 1653, 1659, 1661,
 1678, 1682, 1710, 1746, 1804, 1809, 1815,
 1821, 1861, 1888, 1889, 1908, 1909, 1936,
 1959, 1960, 1965, 1966, 1968, 1971, 1981,
 1987, 2018, 2085, 2086, 2087, 2088, 2127,
 2201, 2202, 2203, 2204, 2664, 2737, 2739,
 2746, 2748, 2749, 2750, 2779, 2784, 2785,
 2814, 2823, 2832, 2833, 2835, 2839, 2849,
 2851, 2853, 2854, 2856, 2862, 2863, 2864,
 2866, 2879, 2880, 2881, 2882, 2887, 2893,
 2894, 3017, 3031, 3032, 3035, 3049, 3054,
 3056, 3057, 3063, 3069, 3089, 3093, 3132,
 3135, 3143, 3146, 3152, 3155, 3156, 3157,
 3160, 3173, 3176, 3177, 3180, 3194, 3195,
 3203, 3204, 3212, 3213, 3214, 3215, 3216,
 3217, 3220, 3228, 3236, 3244, 3250, 3253,
 3260, 3263, 3279, 3280, 3283, 3284, 3287,
 3291, 3302, 3305, 3310, 3324, 3327, 3328,
 3331, 3345, 3346, 3354, 3355, 3363, 3364,
 3365, 3366, 3367, 3368, 3369, 3370, 3378,
 3379, 3385, 3393, 3401, 3409, 3415, 3422,
 3429, 3436, 3451, 3454, 3455, 3458, 3462

21

J

118, 138, 141, 441, 454, 455, 654, 696,
 697, 698, 699, 716, 717, 718, 719, 730,
 732, 738, 748, 750, 756, 786, 884, 929,
 946, 947, 951, 952, 960, 979, 1088, 1131,
 1151, 1154, 1155, 1156, 1180, 1387, 1466,
 1467, 1472, 1473, 1494, 1534, 1563, 1654,
 1655, 1660, 1661, 1678, 1746, 1861, 1873,
 1874, 1875, 1876, 1877, 1883, 1893, 1894,
 1899, 1903, 1913, 1914, 1919, 1936, 1965,
 1966, 1968, 1971, 2664, 2738, 2739, 2747,
 2748, 2749, 2781, 2782, 2783, 2784, 2785,
 2786, 2789, 2792, 2794, 2796, 2799, 2804,
 2806, 2809, 2811, 2815, 2833, 2835, 2839,
 2851, 2853, 2856, 2880, 2882, 2884, 2887,
 2894, 2952, 2953, 3055, 3056, 3057, 3063,
 3097, 3098, 3102, 3106, 3107, 3108, 3113,
 3116, 3117, 3156, 3157, 3176, 3178, 3179,
 3180, 3215, 3216, 3218, 3219, 3220, 3226,
 3227, 3228, 3234, 3235, 3236, 3242, 3243,
 3244, 3250, 3252, 3253, 3260, 3262, 3263,
 3280, 3283, 3284, 3287, 3291, 3305, 3306,
 3307, 3309, 3310, 3327, 3329, 3330, 3331,
 3366, 3368, 3379, 3380, 3383, 3384, 3385,
 3391, 3392, 3393, 3399, 3400, 3401, 3407,

		3408, 3409, 3415, 3419, 3422, 3429, 3433, 3436
22	K	118, 134, 135, 140, 141, 142, 174, 189, 190, 194, 195, 229, 260, 261, 262, 266, 267, 275, 276, 284, 285, 293, 294, 302, 303, 311, 312, 320, 321, 329, 330, 338, 339, 347, 348, 382, 398, 399, 403, 404, 411, 412, 441, 451, 455, 459, 461, 551, 556, 557, 654, 671, 673, 676, 683, 688, 692, 696, 697, 698, 699, 707, 711, 716, 717, 718, 719, 731, 732, 749, 750, 884, 901, 902, 907, 908, 909, 929, 969, 970, 971, 972, 988, 989, 990, 991, 1088, 1107, 1108, 1111, 1112, 1128, 1132, 1133, 1153, 1289, 1294, 1295, 1310, 1387, 1455, 1481, 1482, 1522, 1523, 1586, 1587, 1591, 1592, 1645, 1768, 1769, 1773, 1774, 1861, 1936, 2233, 2282, 2283, 2287, 2288, 2316, 2317, 2321, 2322, 2367, 2403, 2404, 2414, 2419, 2424, 2429, 2434, 2439, 2444, 2449, 2454, 2459, 2469, 2470, 2480, 2485, 2490, 2495, 2500, 2505, 2510, 2515, 2520, 2525, 2664, 2747, 2783, 2788, 2789, 2792, 2795, 2796, 2799, 2806, 2807, 2809, 2811, 2815, 2848, 2852, 2853, 2854, 2855, 2883, 2884, 2887, 2895, 2896, 2903, 2905, 2908, 2912, 2914, 2917, 2919, 2953, 2954, 2956, 3076, 3078, 3081, 3102, 3105, 3107, 3108, 3112, 3113, 3116, 3117, 3177, 3179, 3180, 3281, 3283, 3287, 3291, 3328, 3330, 3331, 3366, 3368, 3418, 3419, 3432, 3433, 3452, 3454, 3458, 3462, 3636, 3715, 3716, 3720, 3721, 3757, 3758, 3762, 3763, 3889, 3955, 3956, 3976, 3977
23	CENSUSVALUES	120, 141, 145, 176, 195, 198, 235, 267, 270, 276, 279, 285, 288, 294, 297, 303, 306, 312, 315, 321, 324, 330, 333, 339, 342, 348, 351, 384, 404, 407, 412, 415, 933, 967, 972, 986, 991, 1271, 1286, 1295, 1427, 1455, 1457, 1645, 1647, 2252, 2283, 2286, 2288, 2290, 2317, 2320, 2322, 2324, 2379, 2414, 2416, 2419, 2421, 2424, 2426, 2429, 2431, 2434, 2436, 2439, 2441, 2444, 2446, 2449, 2451, 2454, 2456, 2459, 2461, 2480, 2482, 2485, 2487, 2490, 2492, 2495, 2497, 2500, 2502, 2505, 2507, 2510, 2512, 2515, 2517, 2520, 2522, 2525, 2527, 3673, 3716, 3719, 3721, 3723, 3758, 3761, 3763, 3765
24	MAXCRG	120, 121, 176, 177, 235, 236, 384, 385, 933, 934, 1286, 1427, 2252, 2379, 3673
25	FORESIGHTADJ	121, 135, 142, 177, 190, 195, 236, 262, 267, 276, 285, 294, 303, 312, 321, 330, 339, 348, 385, 399, 404, 412
26	EUINDEX	133, 151, 188, 206, 257, 354, 397, 417, 2739, 3069, 3089
27	SEC	133, 145, 151, 188, 193, 198, 206, 257, 270, 279, 288, 297, 306, 315, 324, 333, 342, 351, 354, 397, 407, 415, 417, 734, 752, 971, 972, 990, 991, 1029, 1032, 1035, 1038, 1457, 1647, 2286, 2290, 2320, 2324, 2416, 2421, 2426, 2431, 2436, 2441, 2446, 2451, 2456, 2461, 2482, 2487, 2492, 2497, 2502, 2507, 2512, 2517, 2522, 2527, 3057, 3102, 3106, 3107, 3108, 3113, 3116, 3117, 3606, 3607, 3608, 3609, 3719, 3723, 3761, 3765
28	COM	133, 145, 151, 752, 990, 991, 1032, 2286, 2290, 2320, 2324, 2730, 3607, 3719, 3723, 3761, 3765
29	NCENSUSREG	134, 140, 189, 194, 260, 266, 275, 284, 293, 302, 311, 320, 329, 338, 347, 398, 403, 411, 451, 728, 746, 950, 959, 966, 978, 985, 1294, 1454, 1644, 2282, 2287, 2316, 2321, 2413, 2418, 2423, 2428, 2433, 2438, 2443, 2448, 2453, 2458, 2479, 2484, 2489, 2494, 2499, 2504, 2509, 2514, 2519, 2524, 2736, 3105, 3112, 3715, 3720, 3757, 3762
30	XQELCM	135, 1645
31	CMNUMBLDG	137, 453
32	CMNUMSERV	138, 454
33	ENDUSECONSUMP	141, 455

34	ELINDEX	141, 455
35	COMDEMADJFAC	142, 459, 461
36	LFINUM	144, 145, 197, 198, 269, 270, 272, 278, 279, 281, 287, 288, 290, 296, 297, 299, 305, 306, 308, 314, 315, 317, 323, 324, 326, 332, 333, 335, 341, 342, 344, 350, 351, 353, 406, 407, 414, 415, 591, 1026
37	LOADFOREC	145, 198, 270, 272, 279, 281, 288, 290, 297, 299, 306, 308, 315, 317, 324, 326, 333, 335, 342, 344, 351, 353, 407, 415, 592, 1054, 1334, 1335
38	DSMIVAL	145, 198, 270, 279, 288, 297, 306, 315, 324, 333, 342, 351, 407, 415, 972, 991, 1293, 1295, 1457, 1647, 2286, 2290, 2320, 2324, 2416, 2421, 2426, 2431, 2436, 2441, 2446, 2451, 2456, 2461, 2482, 2487, 2492, 2497, 2502, 2507, 2512, 2517, 2522, 2527, 3719, 3723, 3761, 3765
39	UNCONFA	146, 199, 408, 416, 1458, 1648, 2304, 2307, 2339, 2342, 3078, 3738, 3741, 3780, 3783
40	IND	188, 193, 198, 206, 1035, 2731, 3608
41	XQELIN	190
42	NEUSES	193, 1131, 3054, 3063
43	QELIN	195
44	HTRCON	230, 245, 267
45	COOLCN	230, 246, 276
46	H2OCON	230, 247, 285
47	REFCON	230, 248, 294
48	FRZCON	230, 249, 303
49	LTCON	230, 250, 312
50	APCON	230, 251, 321
51	RSFLCN	231
52	CKCON	232, 252, 330, 348
53	DRYCON	232, 253, 339
54	SHTCON	232, 254
55	TGN	233, 258, 271, 272, 280, 281, 289, 290, 298, 299, 307, 308, 316, 317, 325, 326, 334, 335, 343, 344, 352, 353, 1416, 1486, 1505, 1527, 1546
56	HTCN	245
57	NYRRD	245, 246, 247, 248, 249, 250, 251, 252, 253, 254
58	NEFHTR	245
59	NCRRD	245, 246, 247, 248, 249, 250, 251, 252, 253, 254
60	CLCN	246
61	NEFCOO	246
62	HWCN	247
63	NEFWHR	247
64	RFCN	248
65	FZCN	249
66	LTC	250
67	APC	251
68	CKCN	252
69	NEFSTO	252
70	DRYCN	253
71	NEFDRY	253
72	SHC	254
73	NEFSHT	254
74	RES	257, 270, 279, 288, 297, 306, 315, 324, 333, 342, 351, 354, 734, 971, 972, 1029, 1457, 1647, 2416, 2421, 2426, 2431, 2436, 2441, 2446, 2451, 2456, 2461, 2482, 2487, 2492, 2497, 2502, 2507, 2512, 2517, 2522, 2527, 2729, 3606
75	XQELRS	261, 262, 1455
76	ELHEATX	267
77	UNCONFAR	270, 279, 288, 297, 306, 315, 324, 333, 342, 351, 2463, 2529
78	RESTGDEM	272, 281, 290, 299, 308, 317, 326, 335, 344, 353, 1505, 1546
79	ELCOOLX	276
80	ELWHEAX	285
81	ELSTOVX	330, 339
82	ELSECHX	348
83	TRA	397, 407, 415, 417, 1038, 2732, 3609
84	XQELTR	399
85	TRQLDV	404
86	ELLDVHX	404
87	TRQRILR	412
88	ELRAILX	412
89	QELCM	459

90	STR1	467, 477, 489, 491, 497, 504, 512, 520, 527, 529
91	STR2	467, 478, 490, 492, 500, 507, 515, 523, 527, 529
92	L1	479, 489, 496, 503, 511, 519, 526
93	L2	480, 490, 499, 506, 514, 522, 526
94	B1	481, 491, 493, 495, 496, 499, 511, 514
95	B2	482, 492, 494, 495, 503, 506, 518, 519, 522
96	DSMCMP	486, 488, 527, 529, 2681, 2789, 2796, 2884, 2905
97	NERCVALUES	534, 548, 557, 659, 732, 734, 750, 752
98	CENSUSDIV	534, 549, 557
99	SECTOR	534, 550, 557, 1271, 1288, 1295, 1300, 1313, 1344, 1346, 1349, 2618, 2629, 2635, 2637, 3597
100	DSMCVAL	555, 557, 734, 752
101	MAPPNTOC	557, 3107
102	BASEYRLD	592, 1050, 1054, 1059, 3078
103	SYSTEMLOAD	594, 1120, 1337
104	BASEYRSYSLD	594, 1061
105	ECPLASTYEARINDEX	618, 620, 1499, 1540, 1572, 1603, 1607, 1684, 1724, 1755, 1786, 1790
106	ECP\$XPH	618, 1393, 1408, 1452, 1466, 1642, 1654, 3825, 3838, 4024, 4037
107	I1	654, 715, 719
108	N	654, 727, 737, 738, 739, 745, 755, 756, 757, 786, 857, 858, 859, 860, 884, 929, 958, 961, 962, 977, 980, 981, 1088, 1406, 1478, 1666, 1936, 2233, 2367, 2664, 2781, 2812, 2813, 2880, 2895, 2899, 2912, 2914, 2917, 2919, 3636, 3889
109	NOPT	654, 729, 736, 742, 747, 754, 760, 1380, 1471, 1472, 1659, 1660
110	C	654, 728, 734, 742, 746, 752, 760, 1387, 1454, 1455, 1644, 1645, 2367, 2413, 2414, 2418, 2419, 2423, 2424, 2428, 2429, 2433, 2434, 2438, 2439, 2443, 2444, 2448, 2449, 2453, 2454, 2458, 2459, 2479, 2480, 2484, 2485, 2489, 2490, 2494, 2495, 2499, 2500, 2504, 2505, 2509, 2510, 2514, 2515, 2519, 2520, 2524, 2525
111	YY	654, 689, 690, 692, 708, 709, 711, 1387, 1577, 1578, 1580, 1759, 1760, 1762
112	CYR1	655, 679, 689, 708, 1419, 1576, 1577, 1758, 1759
113	OPTI	656, 687, 688, 690, 692, 706, 707, 709, 711, 930, 965, 967, 970, 973, 984, 986, 989, 992, 1381, 1473, 1474, 1475, 1476, 1480, 1482, 1521, 1523, 1561, 1580, 1613, 1630, 1661, 1662, 1663, 1664, 1668, 1670, 1705, 1707, 1744, 1762, 1794, 1813, 2220, 2240, 2276, 2279, 2281, 2313, 2315, 2354, 2372, 2402, 2404, 2405, 2468, 2470, 2471, 3615, 3645, 3705, 3712, 3714, 3754, 3756, 3846, 3869, 3896, 3954, 3956, 3975, 3977, 4045
114	DELTA	657, 734, 735, 739, 752, 753, 757
115	PPRAMP	657, 690, 691, 692, 709, 710, 711, 1403, 1578, 1579, 1580, 1760, 1761, 1762
116	RSHIFT	658, 688, 692, 699, 732
117	MAXNRG	658, 659, 671
118	MAXRDSMO	658, 675, 687, 730, 934, 951, 965, 984, 2255, 3676, 3923
119	CSHIFT	658, 707, 711, 719, 750
120	MAXCDSMO	658, 672, 706, 748
121	ROPCHOICE	673, 692, 698, 711, 732, 1580
122	COPCHOICE	676, 718, 750, 1762
123	DSMROPTIONRAMP	690, 1476, 1613, 3217
124	NRPROG	695, 715, 1465, 1682, 1710, 1804, 1809, 1815, 1821, 1882, 1959, 3129, 3132, 3218
125	RPROGDEFN	696, 1471, 1899
126	RPROGDEF	697, 698, 699, 1473, 1494, 1534, 1563, 1874, 1876, 1894
127	DSMPRCHOICE	699, 719
128	DSMCOPTIONRAMP	709, 1664, 1794, 3370
129	NCPROG	714, 1653, 1902, 1959, 3140, 3143, 3383
130	CPROGDEFN	716, 1659, 1919
131	CPROGDEF	717, 718, 719, 1661, 1678, 1746, 1875, 1877, 1914
132	DSMROPTIONINDEX	738, 962
133	DSMRFRACOPTIONMARKET	739
134	DSMROPTIONSNUMBER	742, 960

135	DSMCOPTIONINDEX	756, 981
136	DSMCFRACOPTIONMARKET	757
137	DSMCOPTIONSNUMBER	760, 979
138	ONE	773, 855, 1082, 1146, 1171, 1257, 2128, 2132, 2135, 2136, 2149, 2198, 2200, 2204, 2206, 2207, 2208, 2211, 2212, 2214, 2658, 2987
139	SYLOAD1	779, 806, 810, 822, 834, 835, 838, 839, 847, 848, 1091, 1143, 1146, 1147, 1177, 1200, 1204
140	MAXRHOUR	779
141	SYLOAD2	780, 810, 814, 822, 1178, 1204, 1209, 1224, 1228, 1236, 1238, 1241
142	MAXHOUR	780, 1091, 1092, 1106, 1115, 1177, 1178, 1316, 1937, 2255, 2256, 3665, 3676, 3677, 3920, 3923, 3924
143	NBLOCKINSEASON	781, 829, 844, 847, 849, 850, 851, 855, 856, 862, 866, 867, 868
144	NBLOCKINLDC	782, 827, 843, 845, 846, 848, 2711, 2997, 3013, 3015, 3018, 3020
145	ISEG	783, 813, 815, 826, 833, 834, 835, 838, 839, 842, 2706, 2970, 2978, 2979, 2980, 2981, 2983, 2984, 2996, 2999, 3002, 3006, 3007, 3008, 3011, 3012, 3015, 3016, 3017, 3021
146	ISEGMENT	784, 832, 836, 840, 851, 2705, 2977, 2982, 2984, 3001
147	ISEASON	785, 828, 832, 858, 859, 860, 863, 864, 865, 866, 867, 868, 2704, 2950, 2951, 2952, 2953, 2955, 2971, 2973, 2975, 2977, 2979, 2987, 2988, 2989, 3000, 3001
148	H	786, 818, 821, 822, 1180, 1225, 1226, 1228, 1236, 1238, 1241, 2233, 2367, 2664, 2813, 2814, 2834, 2913, 2915, 2922, 2927, 2940, 2941, 2942, 2943, 2944, 2964, 2965, 2967, 3636, 3748, 3749, 3750, 3790, 3791, 3792, 3889, 3969, 3970, 3971, 3990, 3991, 3992
149	HN	786, 805, 806, 819, 820, 822, 1187, 1199, 1200
150	NB	786, 856, 857, 858, 859, 860
151	IH1	787, 845, 847
152	IH2	787, 846, 847, 848
153	YCOORDINATE	788, 847, 855, 858
154	MAXEFDB	788, 789, 790, 791
155	BLOCKRANK	789, 850, 859
156	SEGASSIGN	790, 851, 860
157	BLOCKIND	791, 849, 855, 859, 860
158	PEAK	792, 830, 834, 835, 863, 885, 906, 908, 909, 912
159	KNEE	792, 831, 838, 839, 866
160	IPEAK	793, 836, 865
161	IKNEE	793, 840, 868
162	NHOUR	805, 809, 819, 907, 1141, 1146, 1152, 1199, 1203, 1332, 1340, 1345, 1962, 1970, 1972, 2830, 2836, 2837, 2838, 2839, 2844, 2862, 2869, 2935, 2940, 2947, 2964, 3702, 3748, 3749, 3790, 3791, 3945, 3969, 3970, 3990, 3991
163	SYLOAD	806, 1116, 1142, 1143, 1200, 1341, 1963, 1973, 3703, 3750, 3792, 3946, 3971, 3992
164	HOURNUMBEREFD	810, 814, 820, 2838, 2935, 2967
165	NUMSEFD	813, 826, 2996, 2999
166	NFIRSTCALHINSEG	815, 2939, 2944
167	NLASTCALHINSEG	815, 2942, 2947
168	HOURLYWEIGHTS	820, 1153, 1226, 2839, 2967
169	EFDNS	828, 2876, 2879, 2893, 2950, 2971, 3000
170	EFDNUMSEG	832, 2881, 2894, 2900, 2977, 2987, 2988, 3001
171	NFIRSTHINSEG	834, 835, 3006, 3015
172	NLAsthINSEG	838, 839, 3008
173	NUMBEROFBLOCKSINSEG	842, 2980, 2981, 2983, 2991, 3011, 3017
174	HCOORDINATE	845, 846, 848, 3015, 3020
175	EFDLDCYC	858, 863, 866
176	EFDLDCPR	859, 864, 867
177	EFDLDCSA	860, 865, 868
178	M	884, 929, 1088, 1312, 1332, 1340, 1341, 1345, 1346, 1347, 1936, 1962, 1963, 1970, 1972, 1973, 1980, 1981, 1986, 1987, 2233, 2367, 2664, 2895, 2902, 3285, 3286, 3287, 3456, 3457, 3458, 3636, 3702, 3703, 3803, 3804, 3831, 3832, 3839, 3840, 3889, 3945, 3946, 4003, 4004, 4030, 4031, 4038, 4039

179	AVERAGE	886, 900, 902, 904, 905
180	PHOUR	887
181	PDAY	888
182	NUMSEC	897, 1028, 1130, 3045, 3049, 3055, 3093, 3097
183	SECANNUALPEAK	898, 912
184	SECTORLOAD	898, 902, 908, 909, 1108, 1346
185	SYSTEMPEAKHOUR	898, 1148
186	NPEAKH	901, 904, 1155, 2762
187	SYSPEAKHOUR	902, 1156
188	SECANNPEAAVPCP	905
189	SYSTEMLF	914
190	SYSTEMLOADFACTOR	914, 1150
191	REBATE	931, 970, 972, 989, 991, 1411, 1561, 1563, 1629, 1744, 1746, 1813, 2220, 2245, 2347, 2349, 2354, 2377, 2533, 2534, 2535, 2536, 3664, 3797, 3853, 3856, 3858, 3860, 3864, 3869, 3919, 3997, 4051, 4054, 4055, 4057, 4061
192	ANNUALENSAVINGS	932, 970, 972, 989, 991, 2220, 2244, 2275, 2310, 2345, 2347, 2350, 2354, 2376, 2399, 2465, 2531, 2533, 2535, 3654, 3710, 3746, 3788, 3795, 3846, 3856, 3909, 3952, 3967, 3988, 3995, 4045, 4054
193	CV	934, 952, 962, 967, 981, 986
194	MAXSEC	946, 1107, 1111, 2737
195	DSMANNUALCOST	947, 971, 990
196	DSMRNUMUNITCHANGE	962
197	DSMROPTIONCOST	973, 1630, 3216, 4045
198	DSMCSEVDEMCHANGE	981
199	DSMCOPTIONCOST	992, 1813, 3369, 3846
200	FLAG	1015, 1047, 1050, 1052
201	TOTDEMCENSUS	1027
202	NEU	1049, 1053, 1058, 2733, 3067, 3069, 3072, 3078, 3081, 3089
203	ZERO	1082, 1121, 2128, 2131, 2215
204	TD	1089, 1139, 1140, 1142
205	SYSTLO	1090, 1129, 1135, 1140, 1149, 1150
206	HINDEX	1092, 1144, 1146, 1148, 1153, 1156
207	TOTSECLoad	1112, 1135, 1349, 3606, 3607, 3608, 3609
208	NERCLSRN	1121, 3035
209	EURECNUM	1133, 3081
210	NERCTDLOSS	1139, 3032
211	SYSTEMPEAK	1147, 1150
212	TOTSYSTEMLOAD	1149
213	NHOURLYR	1150, 2831, 2835
214	SWITCH	1163, 1181, 1206, 1255
215	AREA	1179, 1218, 1228, 1236, 1238, 1246, 1249
216	H1	1180, 1222, 1224, 1225
217	H2	1180, 1223, 1225
218	W	1180, 1226
219	DH	1182, 1231, 1236, 1238
220	PL	1183, 1224, 1234, 1241
221	SGN	1184, 1220, 1222, 1223, 1229, 1230, 1231, 1232, 1233, 2693, 2860, 2861, 2863, 2864, 2865, 2866, 2869
222	BLK	1185, 1216, 1234, 1236, 1240, 1246, 1247, 1251, 1257
223	BLS	1186, 1217, 1229, 1230, 1231, 1232, 1233, 1239, 1248
224	HOURNUMBER	1204, 1209, 1226, 2837, 2844
225	ECPNUMSG	1208, 1220, 2766, 2779, 2849, 2860
226	ECPSGFH	1209, 1222, 2861, 2866
227	ECPSGLH	1209, 1223, 2864, 2869
228	ECPSGDNB	1229, 2784, 2785, 2851
229	ECPBLOCKX	1230, 1231, 2856
230	ECPSGDBLTYP	1232, 1233, 2784
231	BLOCKHEIGHT	1234, 1236, 1246, 1257, 1263, 1981, 3804, 4004
232	ECPBLWIDTH	1236, 1246, 1264, 2853, 2855
233	ECPNUMBL	1251, 1252, 1262, 1980, 1986, 3803, 3831, 3839, 4003, 4030, 4038
234	BLOCKNUM	1253, 1257, 1264, 1265, 1981, 3804, 4004
235	ECPLDCBH	1263
236	ECPLDCBW	1264
237	SUMSEGWIDTH	1264, 2742, 2824, 2826
238	ECPLDCBS	1265
239	ECPBLSEG	1265, 2854
240	NERCREG	1271, 1287, 1295
241	MAPPCTON	1295, 3102, 3106, 3108, 3113, 3116, 3117
242	NUMREC	1300, 1315, 1331, 1941, 1968, 1969, 3638, 3747, 3748, 3789, 3790, 3891, 3968, 3969, 3989, 3990

243	LFPOINTER	1300, 1314, 1333, 1334, 1335
244	DISTLO	1316, 1332, 1341, 1347, 1937, 1970, 1973, 3665, 3748, 3750, 3790, 3792, 3920, 3969, 3971, 3990, 3992
245	LOAD1	1317, 1334, 1347, 1349
246	LOAD2	1318, 1335, 1337, 1339, 1341
247	LSRNAME	1319, 1331, 1942, 1969, 3666, 3748, 3790, 3921, 3969, 3990
248	NUMBYR	1368, 1394, 1397
249	NYREST1	1369, 1377
250	NYREST2	1369
251	IDENTIF	1374, 1490, 1531, 1675, 1713, 2701, 3283, 3286, 3454, 3457
252	UECC	1375, 1744, 1803, 1805, 1808, 1810, 2249, 2291, 2293, 2310, 2325, 2327, 2345, 3615, 3670, 3724, 3726, 3746, 3750, 3766, 3768, 3788, 3792
253	MAXCTECH	1375, 1414, 3670
254	UEC	1376, 1491, 1532, 1561, 1622, 1623, 1626, 1627, 2378, 2416, 2421, 2426, 2431, 2436, 2441, 2446, 2451, 2456, 2461, 2463, 2482, 2487, 2492, 2497, 2502, 2507, 2512, 2517, 2522, 2527, 2529, 3869, 3902, 3965, 3986
255	NYRESTR	1376, 1378, 1491, 1502, 1511, 1513, 1532, 1543, 1552, 1554, 1610, 3902
256	MAXDMT	1376, 1377, 1378, 3902
257	MKS	1377, 1490, 1508, 1513, 1531, 1549, 1554, 1587, 1592
258	MAXRDT	1377
259	CNS	1378, 1491, 1504, 1505, 1532, 1545, 1546
260	CNSC	1379, 1676, 1688, 1714, 1720, 1728
261	NYRESTC	1379, 1414, 1676, 1686, 1696, 1698, 1714, 1716, 1726, 1736, 1738
262	MAXNTPO	1379, 2702, 2703
263	NTECH	1382, 1480, 1481, 1521, 1522, 1668, 1669, 1705, 1706, 2235, 2279, 2280, 2313, 2314, 2369, 2402, 2403, 2468, 2469, 3640, 3712, 3713, 3754, 3755, 3893, 3954, 3955, 3975, 3976
264	BUILDI	1383, 1474, 1485, 1526, 1662, 1671, 1682, 1710, 1711, 2241, 2276, 2288, 2322, 2373, 2405, 2414, 2419, 2424, 2429, 2434, 2439, 2444, 2449, 2454, 2459, 2471, 2480, 2485, 2490, 2495, 2500, 2505, 2510, 2515, 2520, 2525, 3646, 3705, 3721, 3747, 3763, 3789
265	DECTI	1384, 1475, 1587, 1592, 1663, 1769, 1774
266	TI	1385, 1482, 1483, 1485, 1486, 1497, 1523, 1524, 1526, 1527, 1538, 1670, 1671, 1682, 1707, 1710, 1711, 2236, 2281, 2283, 2284, 2288, 2292, 2295, 2296, 2297, 2299, 2303, 2304, 2306, 2315, 2317, 2318, 2322, 2326, 2330, 2331, 2332, 2334, 2338, 2339, 2341, 2370, 2404, 2406, 2409, 2410, 2412, 2414, 2417, 2419, 2422, 2424, 2427, 2432, 2437, 2442, 2444, 2447, 2449, 2452, 2454, 2457, 2459, 2470, 2472, 2475, 2476, 2478, 2480, 2483, 2485, 2488, 2490, 2493, 2498, 2503, 2508, 2510, 2513, 2515, 2518, 2520, 2523, 2525, 3641, 3714, 3716, 3717, 3721, 3725, 3729, 3730, 3731, 3733, 3737, 3738, 3740, 3744, 3745, 3747, 3756, 3758, 3759, 3763, 3767, 3771, 3772, 3773, 3775, 3779, 3780, 3782, 3786, 3787, 3789, 3894, 3956, 3957, 3961, 3962, 3963, 3964, 3968, 3977, 3978, 3981, 3982, 3983, 3984, 3989
267	TDN	1386, 1483, 1484, 1524, 1525, 2239, 2371, 2406, 2408, 2472, 2474, 3644, 3895, 3957, 3959, 3978, 3980
268	Y	1387, 1453, 1455, 1490, 1491, 1499, 1500, 1501, 1531, 1532, 1540, 1541, 1542, 1572, 1574, 1578, 1600, 1601, 1643, 1645, 1675, 1676, 1684, 1685, 1686, 1688, 1693, 1698, 1713, 1714, 1716, 1718, 1720, 1724, 1725, 1726, 1728, 1733, 1738, 1755, 1760, 1769, 1774, 1782, 1783, 3649, 3817, 3825, 3826, 3904, 4017, 4024, 4025
269	D	1387, 1490, 1507, 1508, 1512, 1513, 1531, 1548, 1549, 1553, 1554, 1675, 1692, 1693, 1697, 1698, 1713, 1717, 1718, 1732, 1733, 1737, 1738, 2233, 2367, 3636, 3889
270	YR	1387, 1452, 1453, 1457, 1500, 1505,

271	YRR	1511, 1541, 1546, 1552, 1642, 1643, 1647, 1685, 1696, 1725, 1736
272	SYR	1387, 1501, 1502, 1504, 1505, 1508, 1513, 1542, 1543, 1545, 1546, 1549, 1554, 1574, 1587, 1592, 1609, 1610, 1622, 1623, 1626, 1627, 3637, 3890, 3948, 3965, 3986
273	SYRIX	1388, 1603, 1604, 1607, 1786, 1787, 1790
274	PYR	1389, 1604, 1613, 1614, 1615, 1623, 1627, 1629, 1632, 1787, 1794, 1795, 1796, 1804, 1809, 1812, 1815, 1940, 1961, 1966, 1971, 1981, 1985, 1987
275	PYRIX	1390, 1607, 1608, 1609, 1614, 1615, 1617, 1618, 1790, 1791, 1795, 1796, 1798, 1799
276	FTO	1391, 1608, 1613, 1614, 1615, 1623, 1627, 1791, 1794, 1795, 1796, 1804, 1809
277	COST	1392, 1519, 1586, 1591, 1621, 1625, 1703, 1768, 1773, 1802, 1807
278	NUMBAPPL	1393, 1467, 1629, 1632, 1655, 1812, 1815, 2409, 2475, 2577, 2578, 2579, 2580, 2581, 2582, 2583, 2584, 2585, 2586, 2587, 2588, 2589, 2590, 2591, 2593, 2594, 2595, 2596, 2597, 2599, 2600, 2601, 2602, 2603, 2606, 2607, 2608, 2610, 2611, 2612, 3961, 3981
279	NUMBAPPLCUM	1394, 1600, 1601, 1614, 1615, 1617, 1618
280	SERVDEM	1396, 1605, 1620, 1623, 1627
281	SERVDEMCUM	1397, 1782, 1783, 1795, 1796, 1798, 1799
282	RMS	1399, 1788, 1801, 1805, 1810
283	RAMP	1400, 1463, 1595, 1652, 1777
284	PRAMP	1401
285	DF	1402, 1476, 1578, 1614, 1615, 1664, 1760, 1795, 1796
286	RATIO	1404, 1606, 1611, 1630, 1789, 1792, 1813
287	NT	1405, 1505, 1508, 1511, 1513, 1546, 1549, 1552, 1554, 1689, 1693, 1696, 1698, 1729, 1733, 1736, 1738
288	NTA	1406, 1477, 1487, 1490, 1491, 1493, 1504, 1505, 1508, 1513, 1519, 1528, 1531, 1532, 1535, 1545, 1546, 1549, 1554, 1564, 1591, 1625, 1634, 1665, 1672, 1675, 1676, 1679, 1688, 1693, 1698, 1703, 1708, 1713, 1714, 1718, 1720, 1728, 1733, 1738, 1747, 1773, 1807, 1817, 2242, 2329, 3648, 3706, 3728, 3770, 3901, 3949, 3960, 3965, 3985, 3986
289	NTB	1407, 1469, 1488, 1493, 1497, 1529, 1535, 1538, 1564, 1637, 1638, 1657, 1673, 1679, 1682, 1709, 1710, 1747, 1820, 1821
290	CNSFR	1407, 1470, 1623, 1627, 1634, 1637, 1658, 1804, 1809, 1817, 1820
291	DSMRMS	1408, 1457, 1505, 1511, 1546, 1552, 1647, 1696, 1736
292	ONEYEARENSAV	1409, 1463, 1652, 2630, 2636, 2638
293	RECN	1412, 1612, 1622, 1626, 1630, 1793, 1803, 1808, 1813
294	SDEM	1413, 1485, 1489, 1490, 1526, 1530, 1531, 1671, 1674, 1675, 1711, 1712, 1713
295	MAXDECT	1414, 1675, 1693, 1698, 1713, 1718, 1733, 1738, 1769, 1774
296	TEFF	1414, 1675, 1713, 1717
297	PASSORFAIL	1415, 2237, 2286, 2291, 2320, 2325, 3642, 3719, 3724, 3761, 3766
298	MMKSF	1417, 1561, 1562, 1744, 1745, 1864, 3615, 3647, 3796, 3852, 3855, 3859, 3863, 3869, 3897, 3996, 4050, 4053, 4056, 4060
299	MMKST	1420, 1585, 1587, 1595, 1767, 1769, 1777
300	TOTNUMBAPPL	1420, 1590, 1592, 1595, 1601, 1772, 1774, 1777, 1783
301	TOTSERVDEM	1421, 1595, 1596, 1597, 1600
302	CURRNA	1422, 1777, 1778, 1779, 1782
303	CURRSD	1423, 1614, 1617, 1620, 1629
304	CURRFR	1424, 1795, 1798, 1801, 1812
305	PERC	1425, 1615, 1618, 1629, 1796, 1799, 1812
306	PAYBACK	1426, 1575, 1580, 1582, 1600, 1601, 1757, 1762, 1764, 1782, 1783
307	DSMROPTIONBUILDT	1463, 1652, 2347, 2533, 2618, 2628, 3125, 3856, 4054
308	DSMROPTIONDECTYPE	1474, 2405, 2471, 3244
309	DSMROPTIONFROMTNUM	1475, 3236
310	DSMROPTIONFROMTECH	1480, 2402, 3214, 3215, 3250, 3954
311	RTECHDMTN	1482, 2404, 3253, 3956
		1483, 1524, 2406, 2472, 3155, 3156,

312	RTECHRRC	3157, 3280, 3957, 3978
313	RTECHDMG	1485, 1526, 3287, 3291 1486, 1527, 2409, 2412, 2414, 2417, 2419, 2422, 2424, 2427, 2432, 2437, 2442, 2444, 2447, 2449, 2452, 2454, 2457, 2459, 2475, 2478, 2480, 2483, 2485, 2488, 2490, 2493, 2498, 2503, 2508, 2510, 2513, 2515, 2518, 2520, 2523, 2525, 3156, 3283, 3961, 3963, 3964, 3981, 3983, 3984
314	RDECTYPN	1490, 1507, 1512, 1531, 1548, 1553, 3191, 3194, 3234
315	LSRINDEX	1497, 1538, 1682, 1710, 1968
316	RTECHLSR	1497, 1538, 3160, 3968, 3989
317	DSMROPTIONTOTNUM	1521, 2468, 3215, 3216, 3260, 3975
318	DSMROPTIONTOTECH	1523, 2470, 3263, 3977
319	DISCFA	1611, 1792, 3810, 3818, 3837, 4010, 4018, 4036
320	DELTAENERGY	1623, 1627, 1804, 1809, 1966, 1971
321	DSMPRCST	1632, 1815
322	NTECHAFFP	1638, 1821, 1960, 1965
323	DSMCOPTIONBUILD	1662, 2276, 3409, 3705
324	DSMCOPTIONDECTYPE	1663, 3401
325	DSMCOPTIONFROMTNUM	1668, 2279, 3365, 3366, 3378, 3379, 3415, 3712
326	DSMCOPTIONFROMTECH	1670, 2281, 3422, 3714
327	CTECHRRC	1671, 1711, 3458, 3462
328	CTECHLSR	1682, 1710, 3310, 3747, 3789
329	CDECTYPN	1692, 1697, 1732, 1737, 3342, 3345, 3399
330	DSMCOPTIONTOTNUM	1705, 2313, 3367, 3368, 3378, 3429, 3754
331	DSMCOPTIONTOTECH	1707, 2315, 3436, 3756
332	IIPROG	1862, 1872, 1874, 1875, 1876, 1877, 1882, 1885, 1894, 1899, 1902, 1905, 1914, 1919
333	IOPTION	1863, 1884, 1885, 1886, 1888, 1889, 1894, 1904, 1905, 1906, 1909, 1914
334	MAXDSMP	1872
335	MAXOPR	1873
336	DSMROPTIONNUMB	1884, 3209, 3212
337	RPROG	1885, 3220
338	DSMROPTIONFYR	1886, 3217
339	BASEYR	1886, 1906, 2592, 2605
340	RRLIST	1888, 1889, 3176, 3177, 3180
341	DSMROPTIONREGION	1888, 1889, 3228
342	DSMCOPTIONNUMB	1904, 3360, 3363
343	CPROG	1905, 3385
344	DSMCOPTIONFYR	1906, 3370
345	CRLIST	1908, 1909, 3327, 3328, 3331
346	DSMCOPTIONREGION	1909, 3393
347	K11	1939, 1958, 1961, 1966, 1971, 1981, 1985, 1987
348	DSMPRLIM	1981, 1987
349	TOSORT	1995, 2004, 2023, 2024, 2025, 2026, 2037, 2041, 2042, 2043, 2044, 2048, 2049, 2050, 2051, 2055, 2056, 2057, 2058, 2063, 2064, 2067, 2069, 2070, 2086, 2089, 2090, 2094, 2095, 2104, 2114, 2140, 2141, 2142, 2143, 2153, 2157, 2158, 2159, 2160, 2164, 2165, 2166, 2167, 2171, 2172, 2173, 2174, 2179, 2180, 2183, 2185, 2186, 2202, 2205, 2206, 2210, 2211
350	JNUMB	1995, 2006, 2021, 2116, 2138, 2139, 2150
351	TOPS	2003, 2075, 2079, 2100, 2113, 2191, 2195, 2216
352	BOTTOM	2003, 2076, 2080, 2101, 2113, 2192, 2196, 2217
353	PIVOT	2004, 2037, 2041, 2043, 2044, 2048, 2050, 2051, 2055, 2057, 2058, 2063, 2067, 2086, 2089, 2094, 2095, 2114, 2153, 2157, 2159, 2160, 2164, 2166, 2167, 2171, 2173, 2174, 2179, 2183, 2202, 2205, 2210, 2211
354	PIVOTI	2005, 2038, 2046, 2047, 2053, 2054, 2060, 2061, 2087, 2096, 2115, 2154, 2162, 2163, 2169, 2170, 2176, 2177, 2203, 2212
355	NUMBER	2007, 2021, 2022, 2034, 2036, 2040, 2048, 2049, 2050, 2052, 2053, 2074, 2080, 2081, 2083, 2085, 2101, 2117, 2137, 2152, 2156, 2164, 2165, 2166, 2168, 2169, 2190, 2196, 2197, 2199, 2201, 2217
356	ITEMP	2008, 2024, 2026, 2027, 2029, 2118, 2141, 2143, 2144, 2146
357	IPART	2009, 2031, 2036, 2039, 2041, 2042, 2043, 2045, 2046, 2055, 2056, 2057, 2059, 2060, 2074, 2075, 2077, 2083, 2084, 2100,

		2119, 2148, 2152, 2155, 2157, 2158, 2159,
		2161, 2162, 2171, 2172, 2173, 2175, 2176,
		2190, 2191, 2193, 2199, 2200, 2216
358	INIT	2010, 2032
359	LIMDEX	2011, 2033, 2075, 2076, 2079, 2080,
		2082, 2098, 2099, 2100, 2101, 2120, 2149,
		2191, 2192, 2195, 2196, 2198, 2214, 2215,
		2216, 2217
360	MEDIAN	2012, 2036, 2037, 2038, 2042, 2044,
		2045, 2047, 2049, 2051, 2052, 2054, 2056,
		2058, 2059, 2061, 2121, 2152, 2153, 2154,
		2158, 2160, 2161, 2163, 2165, 2167, 2168,
		2170, 2172, 2174, 2175, 2177
361	ITOP	2013, 2039, 2066, 2067, 2068, 2069,
		2070, 2071, 2072, 2074, 2077, 2079, 2088,
		2089, 2090, 2091, 2092, 2093, 2094, 2095,
		2096, 2122, 2155, 2182, 2183, 2184, 2185,
		2186, 2187, 2188, 2190, 2193, 2195, 2204,
		2205, 2206, 2207, 2208, 2209, 2210, 2211,
		2212
362	IBOT	2014, 2040, 2062, 2063, 2064, 2065,
		2068, 2069, 2071, 2074, 2076, 2081, 2123,
		2156, 2178, 2179, 2180, 2181, 2184, 2185,
		2187, 2190, 2192, 2197
363	TEMP2I	2015, 2065, 2072, 2124, 2181, 2188
364	TEMP2	2016, 2064, 2070, 2125, 2180, 2186
365	FIRST	2104, 2128, 2135, 2136, 2138, 2140,
		2141, 2142, 2144, 2145, 2148
366	LAST	2104, 2128, 2137, 2138
367	SECOND	2128, 2136, 2140, 2142, 2143, 2145, 2146
368	ZEROTH	2128, 2135, 2209
369	TWO	2128, 2133, 2139, 2152
370	TEN	2128, 2134, 2150, 2199
371	REGION	2220, 2234, 2347, 2354, 2368, 2533
372	CAPFAC	2238, 2290, 2300, 2307, 2324, 2335,
		2342, 3643, 3723, 3734, 3741, 3765, 3776,
		3783
373	INCROPTIONCOST	2243, 2274, 2296, 2303, 2331, 2338,
		2347, 2375, 2398, 2409, 2475, 2533, 3652,
		3707, 3730, 3737, 3772, 3779, 3809, 3856,
		3907, 3950, 3961, 3981, 4009, 4054
374	OMCOST	2246, 2277, 2299, 2306, 2334, 2341,
		2348, 3667, 3708, 3733, 3740, 3775, 3782,
		3847, 3857
375	ELPRICE	2248, 3669
376	DSMROPTLSR	2255, 3676, 3923
377	HLOAD	2256, 3677, 3924
378	TECHEFF	2283, 2317, 3716, 3758
379	CTECHCODE	2283, 2284, 2288, 2292, 2295, 2296,
		2297, 2299, 2303, 2304, 2306, 2317, 2318,
		2322, 2326, 2330, 2331, 2332, 2334, 2338,
		2339, 2341, 3305, 3419, 3433, 3454, 3455,
		3716, 3717, 3721, 3725, 3729, 3730, 3731,
		3733, 3737, 3738, 3740, 3744, 3745, 3758,
		3759, 3763, 3767, 3771, 3772, 3773, 3775,
		3779, 3780, 3782, 3786, 3787
380	CAPACITYFACTOR	2288, 2322, 3721, 3763
381	TECHCOST	2296, 2299, 2303, 2306, 2331, 2334,
		2338, 2341, 3730, 3733, 3737, 3740, 3772,
		3775, 3779, 3782
382	EPRICE	2347, 2533, 3856, 4054
383	CURRUEC	2374, 2407, 2463, 2465, 2473, 2529,
		2531, 3903, 3958, 3965, 3967, 3971, 3979,
		3986, 3988, 3992
384	RTECHDMT	2410, 2476, 3157, 3284, 3962, 3963,
		3964, 3982, 3983, 3984
385	HTUEC	2414, 2480
386	CLUUEC	2419, 2485
387	HWUEC	2424, 2490
388	RFUEC	2429, 2495
389	FZUEC	2434, 2500
390	LTUEC	2439, 2505
391	APPUEC	2444, 2510
392	CKUEC	2449, 2515
393	DRYUEC	2454, 2520
394	SHTUEC	2459, 2525
395	EQUIPLIFE	2554, 2555, 2556, 2557, 2558, 2559,
		2560, 2561, 2562, 2563, 2564, 2565, 2566,
		2567, 2568, 2569, 2570, 2571, 2572, 2573,
		2574, 2575, 2576, 3963, 3964, 3983, 3984
396	T	2664, 2803, 2815, 2823, 2824, 2917, 2919
397	P	2664, 2810, 2811
398	R	2664, 2918, 2919, 2954, 2956

399	S	2664, 2899, 2900, 2920, 2921, 2988, 2990, 3417, 3419, 3421, 3431, 3433, 3435
400	V	2664, 2921, 2922, 2990, 2991, 3135, 3146, 3155, 3305
401	DAFNR	2667, 2746, 2751, 3033, 3079, 3158, 3308
402	RRFNR	2668, 3274, 3275, 3276, 3285
403	CRFNR	2669, 3447, 3448, 3456
404	DAFIL	2670
405	LSRNAM	2671, 3032, 3034, 3077, 3080, 3157, 3159
406	LSRNAME	2671, 3306, 3309
407	MAXCBT	2671, 3306, 3307
408	DBLSRNAME	2672, 2752, 3034, 3080, 3159, 3309
409	MAXREC	2672
410	REG	2673
411	LREG	2674, 3176, 3179, 3327, 3330
412	MAXRLST	2674
413	DUMMYCHAR	2675, 3213, 3364
414	NUM	2676
415	FLAGEU	2677, 3050, 3068, 3070
416	MONNAM	2678, 2781, 2789, 2880, 2884
417	MAXMON	2678, 2682, 2687
418	DTPNAM	2679, 2782, 2796, 2895, 2905
419	MAXDTP	2679, 2688
420	ECPSGDMONTH	2682, 2792, 2806, 2809, 2811, 2815
421	EFDSEDNM	2683, 2880, 2882, 2912, 2952
422	MAXEFDS	2683, 2687
423	DND	2684, 2782, 2794, 2810, 2895, 2903, 2918
424	DNM	2685, 2781, 2786, 2804
425	DNH	2686, 2783, 2812, 2896, 2920
426	EFDSEDMONTH	2687, 2887, 2914, 2917, 2919, 2953
427	DDAY	2688, 2799, 2811, 2908, 2919
428	DHOUR	2689, 2783, 2813, 2896, 2921
429	MAXITV	2689
430	SINDEX	2690, 2898, 2900, 2902, 2922, 2938, 2939, 2941, 2942, 2943, 2944, 2947, 2961, 2962, 2965, 2966
431	ECPSGWIDTH	2691, 2823, 2853
432	MAXECPS	2691
433	X	2692, 2850, 2855, 2856, 3077, 3078
434	PROGNAM	2694, 3213, 3219, 3364, 3384
435	REGLD	2695, 3213, 3227, 3364, 3392
436	DECTD	2696, 3214, 3235, 3365, 3400
437	BULITD	2697, 3214, 3243, 3365, 3408
438	FTECHLSR	2698, 3215, 3252
439	MAXFRT	2698
440	TTECHLSR	2699, 3216, 3262
441	MAXTOT	2699
442	TECHIDENT	2700, 3277, 3286, 3449, 3457
443	MAXRRST	2700
444	FCTC	2702, 3366, 3371, 3372, 3373, 3376, 3380, 3419
445	NUMCTCE	2702, 2703, 3305, 3366, 3368, 3418, 3421, 3432, 3435
446	TCTC	2703, 3368, 3371, 3374, 3375, 3377, 3380, 3433
447	BLOCKNUMBER	2707, 2973, 2975, 2979, 2980, 2982, 3011, 3012
448	DELTABLOCK	2708, 2982, 2987
449	MSEGEFD	2708, 2710
450	NBLOCK	2709, 2972, 2981, 2989, 2992
451	SEGMENTNUMBER	2710, 2984, 2987, 2990
452	DELTAHINSEGMENT	2712, 3012, 3016, 3021
453	TNSEEFD	2712, 2961
454	IH	2713, 2998, 3005, 3006, 3007, 3008
455	SLNM	2715, 3096, 3098
456	SECNAM	2729, 2730, 2731, 2732, 3056, 3098
457	MNUMNR	2735, 3102, 3113, 3117
458	MNUMCR	2736, 3102, 3106, 3108
459	NMONTH	2746, 2747, 2748, 2749, 2750, 2788, 2832, 2883
460	NODAYS	2746, 2747, 2748, 2749, 2807, 2809, 2833, 2915, 2917, 2954
461	WEIGHT	2747, 2815, 2835, 2839, 2956
462	NODAYT	2747, 2750, 2795, 2904
463	NOSEA	2747, 2749
464	MONTHYP	2748
465	IDAYTQ	2748
466	JDAYTP	2749, 2811, 2919
467	SENAME	2749
468	DTNAME	2750, 2796, 2905
469	MONAME	2750, 2789, 2884
470	ECPSGDBLOCK	2784, 2853
471	ECPSGNUM	2814, 2844, 2863

472	EFDNUMBL	2881, 2973, 2989
473	EFDGNUM	2922, 2935, 2938, 2941, 2943, 2965
474	NHOURINREALSEASON	2951, 2955, 2975
475	NHOURINREALSEGMENT	2962, 2966, 2979, 3007, 3012
476	EFDLDCBW	2975, 2979
477	NERCNAM	3032, 3179, 3330
478	SLNAM	3054, 3056
479	EUNAM	3072
480	RPROGCODE	3135, 3219
481	CPROGCODE	3146, 3384
482	RTECHNUMB	3151, 3152, 3251, 3261, 3279
483	RTECHCODE	3155, 3252, 3262
484	RRLISTN	3170, 3173, 3226
485	RRLISTID	3176, 3227
486	RDECTYPID	3195, 3235, 3400
487	RDECTYPIX	3195, 3236
488	RBUILDTN	3200, 3203, 3242, 3282
489	RBUILDID	3204, 3243
490	RBUILDITX	3204, 3244
491	DSMROPTIONCODE	3213
492	CTECHNUMB	3301, 3302, 3416, 3430, 3451
493	CRLISTN	3321, 3324, 3391
494	CRLISTID	3327, 3392
495	CDECTYPID	3346
496	CDECTYPIX	3346, 3401
497	CBUILDTN	3351, 3354, 3407, 3453
498	CBUILDID	3355, 3408
499	CBUILDITX	3355, 3409
500	DSMCOPTIONCODE	3364
501	LINE	3567, 3572, 3574, 3575
502	STAR	3568, 3571, 3572, 3574
503	QELRSN	3606, 3610
504	QELCMN	3607, 3610
505	QELINN	3608, 3611
506	QELTRN	3609, 3611
507	QELASN	3610
508	PROFTHR	3633, 3854, 3886, 4052
509	EQL	3639, 3709, 3744, 3745, 3786, 3787, 3817, 3892, 3951, 3963, 3964, 3983, 3984, 4017
510	IYR	3650, 3826, 3832, 3835, 3838, 3840, 3843, 3905, 4025, 4031, 4034, 4037, 4039, 4042
511	OPTIONBENEFITS	3651, 3808, 3845, 3850, 3851, 3906, 4008, 4044, 4048, 4049
512	ENSAVINGS	3653, 3804, 3832, 3835, 3840, 3843, 3908, 4004, 4031, 4034, 4039, 4042
513	MAXECTB	3653, 3908
514	OPTIONCOST	3655, 3809, 3846, 3850, 3854, 3910, 4009, 4045, 4048, 4052
515	AOPB	3656, 3824, 3832, 3835, 3840, 3843, 3845, 3911, 4023, 4031, 4034, 4039, 4042, 4044
516	PVFA	3657, 3810, 3832, 3835, 3840, 3843, 3912, 4010, 4031, 4034, 4039, 4042
517	PVFB	3658, 3815, 3837, 3840, 3843, 3913, 4015, 4036, 4039, 4042
518	PVFC	3659, 3816, 3818, 3847, 3914, 4016, 4018, 4045
519	BENEFITCOSTRATIO	3660, 3850, 3854, 3915, 4048, 4052
520	STOPDIST	3661, 3699, 3810, 3825, 3826, 3916, 3942, 4010, 4024, 4025
521	TECHLIFE	3744, 3745, 3786, 3787
522	EPAVOID	3832, 3840, 4031, 4039
523	EPRMRGN	3835, 3843, 4034, 4042
524	NTAF	3898
525	NTAL	3900