SDC-92-257
ANL-HEP-TR-92-38

Unix Version of CALOR89 for Calorimeter Applications [1]

ANL-HEP-TR--92-38

DE92 016175

T. Handler

University of Tennessee, Knoxville TN 37996

P.K. Job and L.E. Price

Argonne National Lab, Argonne IL 60439

T.A. Gabriel

Oak Ridge National Lab, Oak Ridge, TN 37831

DISCLAIMER

[1]Work supported in part by US Department of Energy, Contract No. W-31-109-ENG-38, No. DE-AC05-84OR21400 and No. DE-AS05-ER03956

SDC-92-00257

SDC
SOLENOIDAL DETECTOR NOTES

Unix Version of CALOR89 for Calorimeter Applications

T. Handler
Department of Physics and Astronomy
University of Tennessee, Knoxville, Tennessee 37966

P. K. Job, and L. E. Price
High Energy Physics Division
Argonne National Laboratory, Argonne, Illinois 60439

T. A. Gabriel
Engineering, Physics, and Mathematics Division
Oak Ridge National Laboratory, Oak Ridge, Tennessee 37831

12 MAY 1992

1. Introduction

CALOR89[1] is a system of coupled Monte Carlo particle transport computer codes (figure 1) which has been successfully employed for the estimation of calorimeter parameters in High Energy Physics [2][3]. In the past CALOR89 has been running on various IBM machines and on CRAY X-MP at Lawrence Livermore Lab. These machines had non-unix operating systems. In this report we present a UNIX version of CALOR89, which is especially suited for the UNIX work stations. Moreover CALOR89 is also been supplemented with two new program packages which makes it more user friendly. CALPREP is a program for the preparation of the input files for CALOR89 in general geometry[4] and ANALYZ is an analysis package to extract the final results from CALOR89 relevant to calorimeters. This report also provides two script files LCALOR and PCALOR. LCALOR runs CALOR89 sequence of programs and EGS4 for a given configuration sequentially on a single processor and PCALOR concurrently on a multiprocessor unix workstation.

2. Program CALPREP

CALOR89 consists of a number of independent programs like HETC[5], EGS4[6] and MORSE[7] working on the same problem. These programs were developed independently in the past for different applications. Therefore they have different input formats and specifications. This made the input preparation for CALOR89 very difficult and vulnerable to mistakes. It will be helpful to have a single input data file for a given run which controls the generation of all the input specifications for different programs in CALOR89. This will avoid inconsistent and redundant input to various programs. The legibility of the input data is also improved by providing a namelist input file.

CALPREP is a program which reads a single input file, CALPREP.INP consisting of information on incident particle and the configuration. In addition to this CALPREP also requires '*.MOR' and '*.HETC' files in the same directory, where '*' is the pegs media name. These are input media specifications for a given material

for HETC and MORSE. These files for the elements and compounds commonly in use must be prepared only once like the pegs cross section file. CALPREP generates six output files which serve as input files to various CALOR89 programs. It also generates the geometry file, in general geometry format[4], which is read by all the CALOR89 programs. This saves the problem of preparing seven different input files into just one file. Figure 2 gives the schematic diagram of the CALPREP program. Appendix A and B provides the listing of the CALPREP.INP and the .SCRIPT files. The input parameters are explained in detail in Appendix A. Table I gives the complete set of input and output files called by each of the CALOR89 programs. The '.bin' files are the cross section data bases. The output print files of all programs are denoted as '.ppr'.

3. Program ANALYZ

ANALYZ is a program to combine the results of various CALOR89 programs and EGS4. This reads a user provided input file and five output files from CALOR89 and EGS4. The output of ANALYZ is a print file called ANALYZ.OUT and a histogram file for PAW called ANALYZ.HST. ANALYZ provides information on several parameters of interest for the calorimeter designers. The package provides information on compensation characteristics and hadronic and electromagnetic resolutions as a function of shower integration time. It also gives the shower depth profiles for both the hadronic and electromagnetic showers. ANALYZ can be recoded for user specific applications. In the present form it gives a near complete set of information for a given configuration. Figure 3 gives the schematic diagram of the ANALYZ package. Appendix C provides information on ANALYZ.INP.

4. Script Programs LCALOR and PCALOR

LCALOR is a script program which It runs the entire sequence of CALOR89 program for hadrons and EGS4 for electrons for a given energy and configuration without the users intervention and analyses the results using the program ANALYZ. The input files generated by CALPREP and ANALYZ.INP must be in the

same directory from which LCALOR is submitted. In LCALOR, the appropriate pathnames for the executable programs of CALOR89 (exec.*) must be provided once, depending upon the installation. For a given installation LCALOR need not be changed afterwards. LCALOR also requires a file called lightm.dat in the same directory which is the light curves for law energy neutrons read by MORSE. Figure 4 gives the schematic diagram of running CALOR89 and EGS4 using LCALOR. Appendix D gives a listing of LCALOR script program.

The sequence of CALOR89(fig.1) shows that once HETC88 is done for a given number of hadron events, SPECT89, EGS4 and MORSE can be run concurrently on the hadronic, electromagnetic and neutronic components of the hadron shower. EGS4 can also run concurrently for pure electromagnetic showers created by the incident electrons. With the advent of multiprocessor machines this became possible. PCALOR utilizes this property to speed up CALOR89. In this way on a multiprocessor workstation CALOR89 programs can be made to run concurrently as long as the HETC output file is on the disk. As in LCALOR the pathnames of the files in PCALOR must be changed once for a given installation. The input files including the geometry must be in the same directory as PCALOR. Figure 5 gives the schematic diagram of running CALOR89 using PCALOR. Appendix E gives the listing of the PCALOR script program.

5. Summary

In this report we have presented a unix version of CALOR89. The difficult problem of preparing several input files for the different programs of CALOR89 has been solved by CALPREP. An analysis program ANALYZ is included which gives many calorimeter parameters of interest. Two script programs are written for running CALOR89 sequentially on single processor and concurrently on multiprocessor unix workstations.

References

[1] T.A. Gabriel et al.,"CALOR, A Monte carlo program package for the design and analysis of the calorimeter systems", ORNL/TM 5619 (1977).

[2] T.Handler et al.,"CALOR89 Calorimeter simulations, Benchmarking and design calculations", Proc. Symp.on Detector Research and Development for SSC, FortWorth, Edited by T.Dombeck et al., World Scientific(1990)608.

[3] P.K. Job et al.,"Comparison of CALOR89 model predictions with scintillator plate calorimeter data", Nucl.Instr.and Meth., A309(1991)60.

[4] HETC User Manual, RSIC code collection, ORNL-CCC-178C.

[5] T.W. Armstrong and K.C.Chandler, HETC- A High Energy Transport Code, Nucl.Sci.and Engg.49(1972)110.

[6] W. R. Nelson, H.Hirayama and D.W.O.Rogers, EGS4 Code System, SLAC 265(1985).

[7] M.B. Emmett and N.M. Greene, Morse Code System, ORNL/TM-3706 (1973).

Table I

List of Input and Output Files of CALOR89 Programs

Program Name	Input Files	Output Files
HETC exec.hetc	hetch.inp *.geo cascade.bin evapor.bin	*.dat hetch.ppr
SPECT exec.spect	spect.inp *.dat kb.dat	s*.dat spect.ppr
EGSPREPH exec.egsprep	egspreph.inp *.dat	egspreph.out egspreph.ppr
EGSH exec.egs	egsh.inp egspreph.out pegs.dat *.geo	e*.dat egsh.ppr
MORSEH exec.morse	morseh.inp *.dat *.geo lightm.dat xsmorse.bin	m*.dat mgams.out morseh.ppr
MEGS exec.egs	megs.inp mgams.out pegs.dat *.geo	em*.dat megs.ppr
EGSE exec.egs	egse.inp (x)e.dat pegs.dat *.geo	e*e.dat egse.ppr

List of Figures

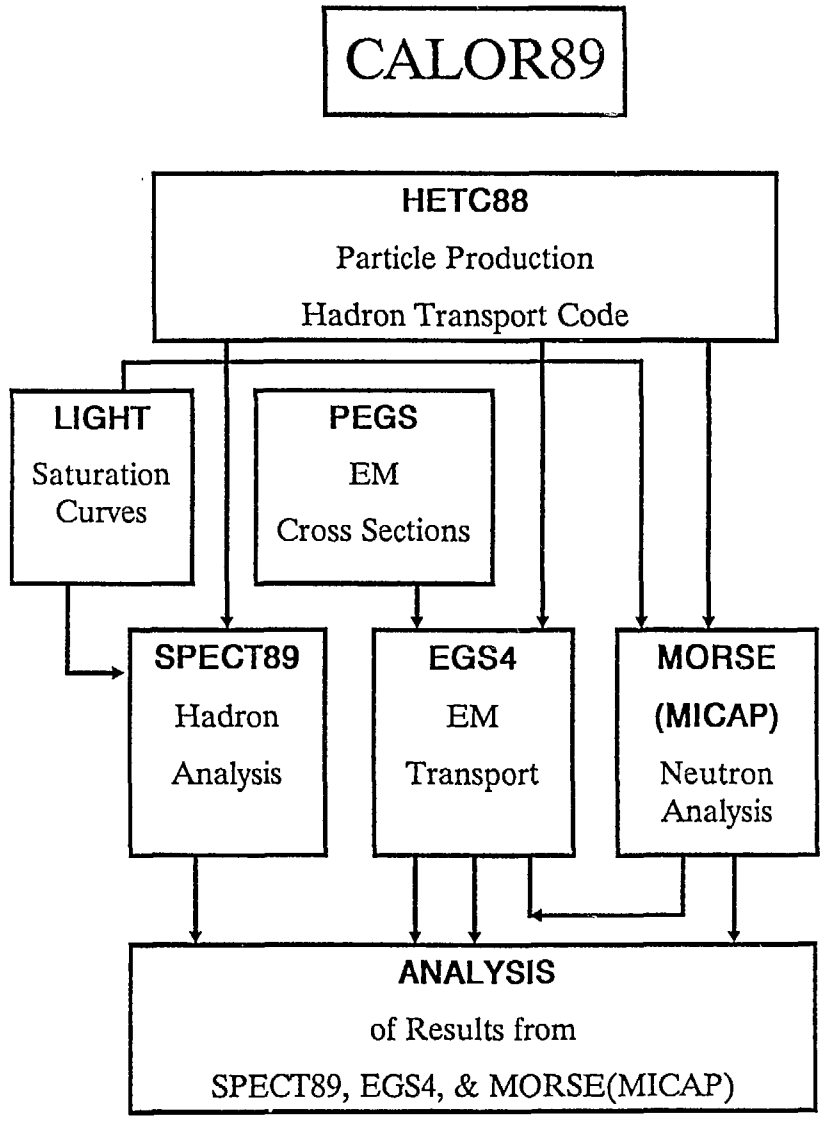
CALOR89
HETC88
Particle Production
Hadron Transport Code
LIGHT
Saturation
Curves
PEGS
EM
Cross Sections
SPECT89
Hadron
Analysis
EGS4
EM
Transport
MORSE
(MICAP)
Neutron
Analysis
ANALYSIS
of Results from
SPECT89, EGS4, & MORSE(MICAP)

Figure 1

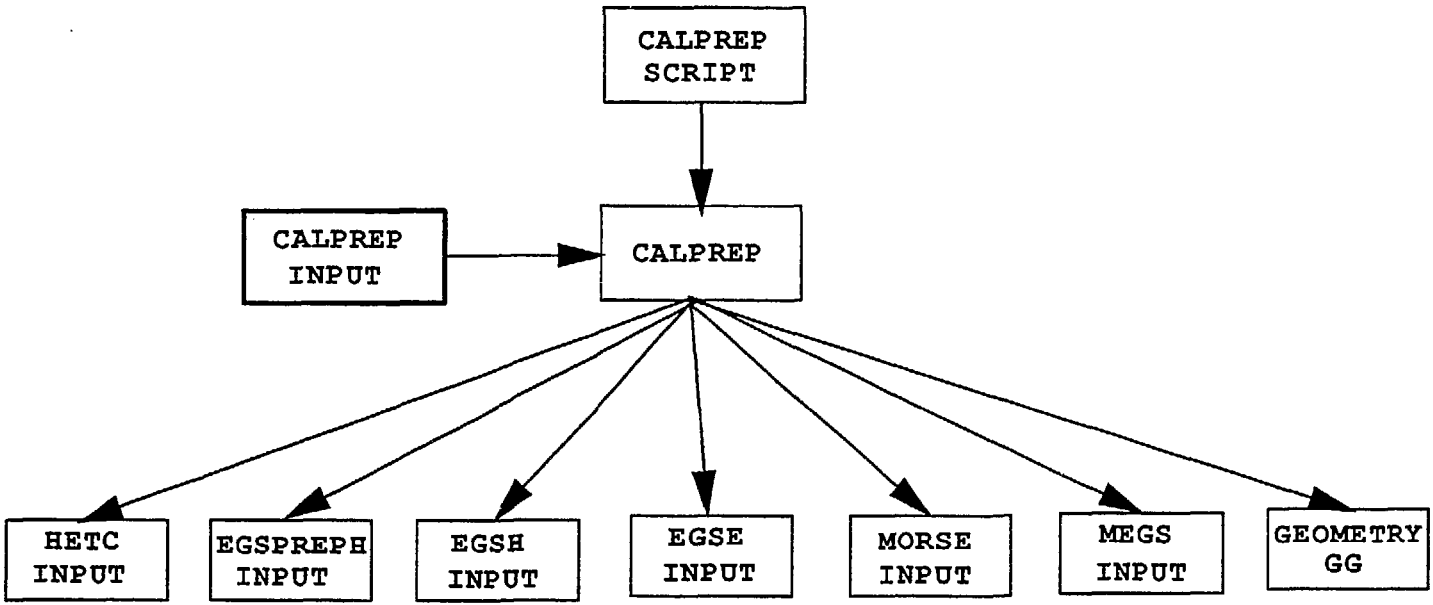
CALPREP
SCRIPT
CALPREP
INPUT
CALPREP
HETC
INPUT
EGSPREPH
INPUT
EGSH
INPUT
EGSE
INPUT
MORSE
INPUT
MEGS
INPUT
GEOMETRY
GG

Figure 2

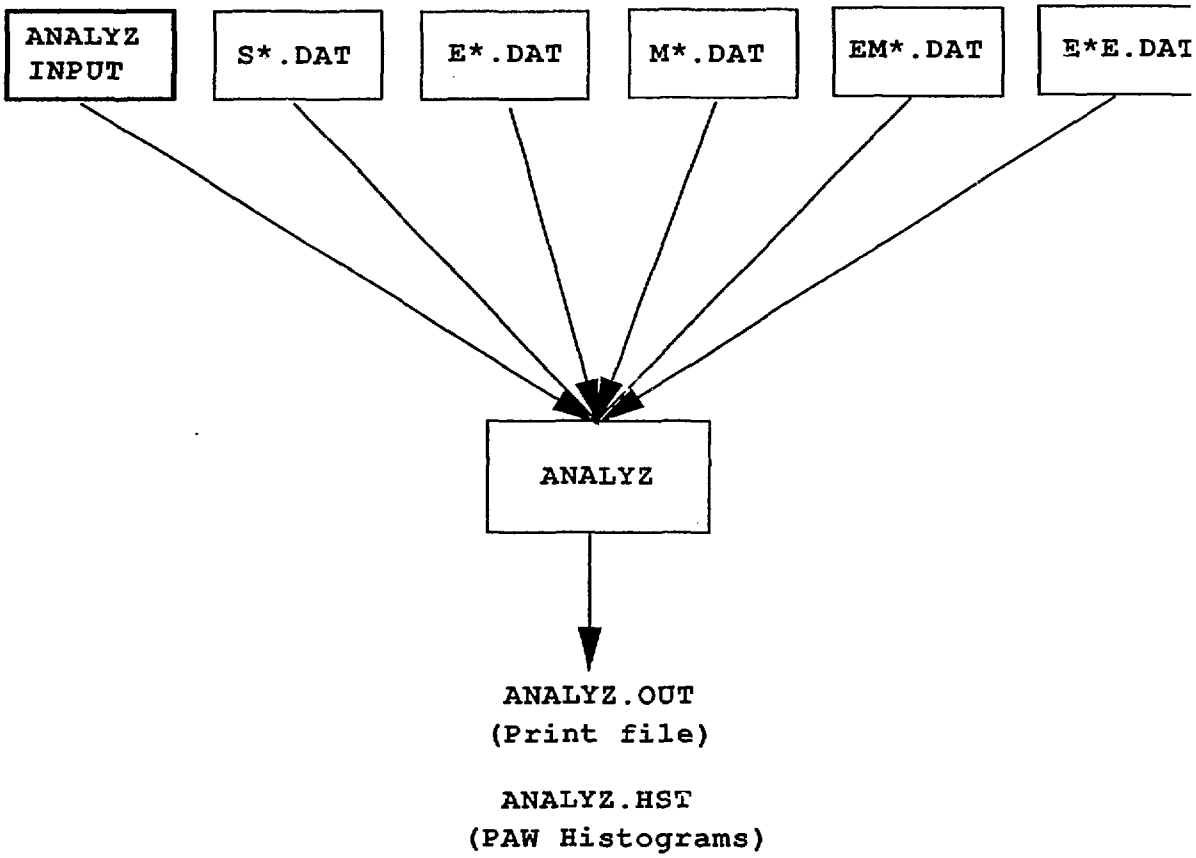
ANALYZ
INPUT
S*.DAT
E*.DAT
M*.DAT
EM*.DAT
E*E.DAT
ANALYZ
ANALYZ.OUT
(Print file)
ANALYZ.HST
(PAW Histograms)

Figure 3

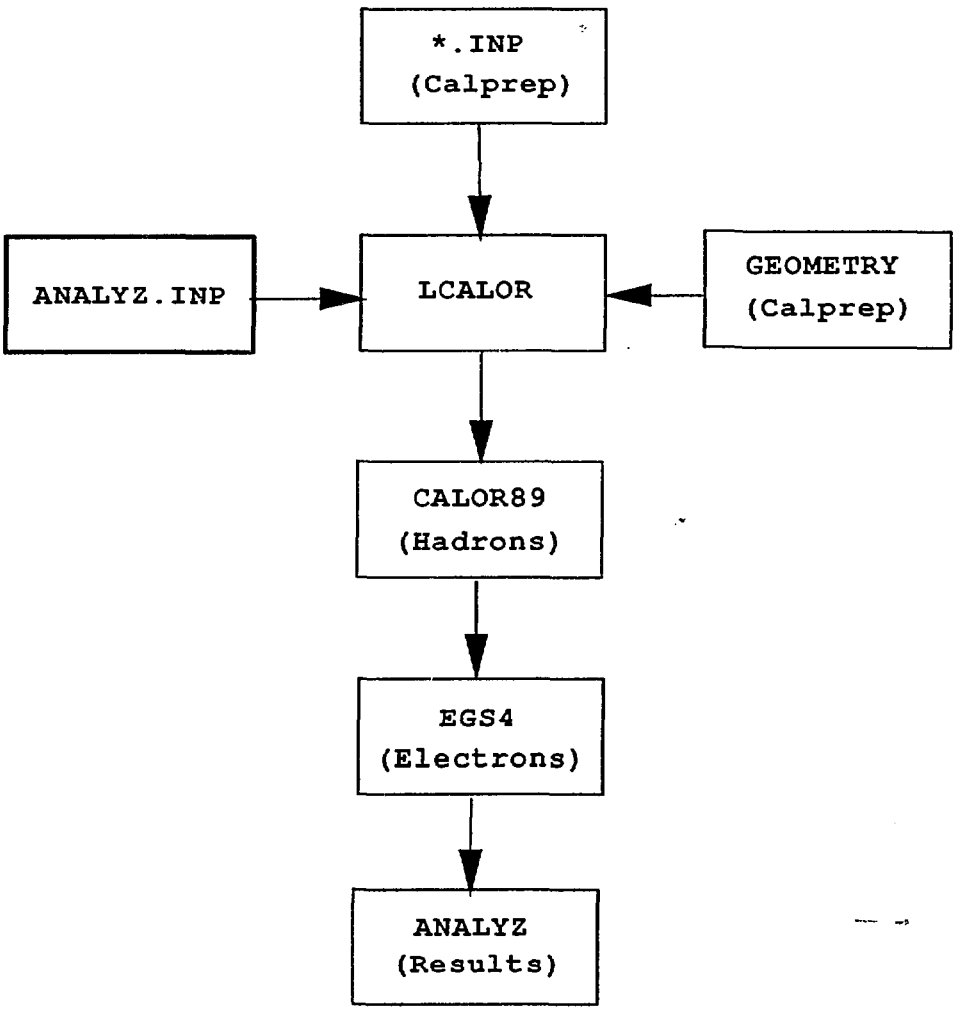
*.INP
(Calprep)
ANALYZ.INP
LCALOR
GEOMETRY
(Calprep)
CALOR89
(Hadrons)
EGS4
(Electrons)
ANALYZ
(Results)

Figure 4

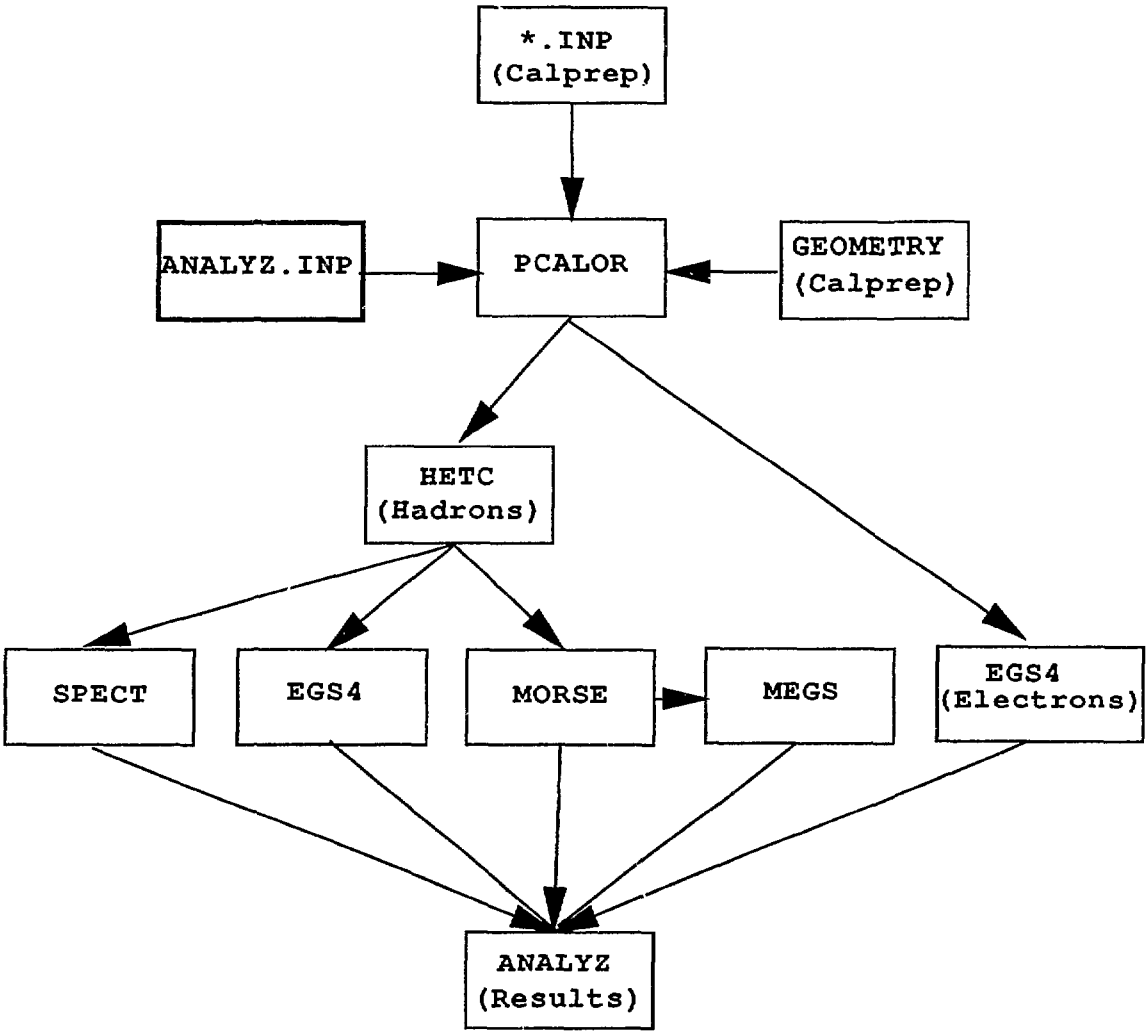
*.INP
(Calprep)
ANALYZ.INP
PCALOR
GEOMETRY
(Calprep)
HETC
(Hadrons)
SPECT
EGS4
MORSE
MEGS
EGS4
(Electrons)
ANALYZ
(Results)

Figure 5

APPENDIX A

```
Data first needed for geometry:
(Material is assumed to begin at z=0 and extend to z>0)
First card below is title (up to 80 characters)
'    10 GeV    3 Module test configuration'
3                    Number of materials(lead,plastic and iron)
'PB'    12.83        Names and dE/dx
'PL'     2.01              of
'FE'    11.65        materials (names must agree with PEGS file)
'PL'                 Name of sensitive material
-50.  50.            xmin, xmax
-50.  50.            ymin, ymax
3                    Number of depth segments (max=5)
36                   # cells in first segment
28                   # cells in 2nd segment
17                   # cells in 3rd segment
2                    # materials in 1st seg unit cell (max=10)
'PB'     0.32          First segment: first material, plate thickness (cm)
'PL'    0.25                        2nd material, plate thickness
2                    # materials in 2nd seg unit cell
'PB'     2.10          Second segment: first material, thickness (cm)
'PL'    0.25                         2nd material
2                    # materials in 3rd seg unit cell
'FE'     5.08          Third segment: first material
'PL'    0.25                        2nd material
----------
Data first needed for HETC:
'test.geo'            Geometry input file
'test.dat'            HETC binary output file (large)
37591                Random number seed for HETC
10140.               Total Energy of incoming particle (MeV)
500                  Number of batches (histories)
----------
Data first needed for SORS:
10000.               Kinetic energy (without mass term)
15000.               ESCALE (MeV)
4.                   Particle type (0=p, 1=n, 2=pi+, 4=pi-, 5=mu+. 6=mu-)
0.                   x |
0.                   y + Source position (cms)
0.01                 z |
0.                   u  |
0.                   v  + Source direction cosines
1.                   w  |
----------
Data first needed for SPECT:
1.                   Saturation: 0=linear 1=nonlinear (run LIGHT for curves)
'stest.dat'           spect output file
'/Net/sgi1/sdc2/pkj/spect89/kb131.dat'          light curves
13793                Random mumber seed for SPECT
----------
Data first needed for EGSPREPH
0                    0=EGS used with HETC, 1=Standalone EGS
0                    0= No low E gammas, -1=l.e. and fission, 1=l.e. not fission
10                   # of source particles printed
13793                Random number seed for EGSPREPH
'egspreph.out'       egsprep output file
35569                egsh random number
'/Net/sgi1/sdc2/pkj/pegs/pbfeal3.p'    PEGS cross section file
'etest.dat'           egsh output file
35569                egse random number
'/Net/sgi1/sdc3/pkj/calor/egse/10e.dat'   electron source file
'eteste.dat'          egse (electrons) output file
35569                megs random number
'emtest.dat'          megs output file
----------
Data first needed for MORSE
5000                 Max # neutrons handled within MORSE per batch
```

```
105                    Number of neutron groups
26                     Number of gamma groups
134                    Cross section table length
4                      Number of coefficients for x-section
5                      Number of basic element cross sections read
4                      Number of mix specifications
500.e-9                Time cut
13793                  Random number seed for MORSE
'     46-group cross sections  id '
    1   13   17   21   33
'mtest.dat'            Morse output file
```

APPENDIX B

```
#! /bin/csh
#
cp calprep.inp fort.30
exec.calprep
mv fort.20 test.geo
mv fort.21 hetch.inp
mv fort.23 spect.inp
mv fort.25 egspreph.inp
mv fort.27 egse.inp
mv fort.28 egsh.inp
mv fort.29 megs.inp
mv fort.26 morseh.inp
rm fort.30
```

APPENDIX C

```
         3
        36          28          17
       1.0         1.0         1.0
         1
      10.0
3 module test configuration
Hadron Energy 10 GeV
stest.dat
etest.dat
mtest.dat
emtest.dat
eteste.dat
      10.0
```

card 1 (8i10) number of longitudinal segments in the calorimeter
card 2 (8i10) number of cells in each segment
card 3 (8f10.4) weight factors by which signal to be enhanced
card 4 (8i10) number of CALOR runs to be analysed
card 5 (8f10) maximum hadron/electron energy used for plots
card 6 (a80) general title
card 7 (a80) specific title
The next five cards are the names of the data files for each run given in the order of spect,egs,morse,megs and egse(electrons) output files. For each subsequent runs to be analysed, these file names are given in the same order (a80).
card 13 (8f10) maximum hadron/electron energy per run

APPENDIX D

```
#! /bin/csh -f
# This script will run the CALOR89 sequence of programs in the foreground.

# Set default of runflag to yes
set runflag = "yes"

# Check that the number of arguments used is 0 or 1
if ($#argv > 1) goto USAGE

# the following is the set of files that must exist in the current directory.
set efiles = (hetch.inp egspreph.inp egsh.inp morseh.inp megs.inp lightm.dat)

# the following lists the set of files that must exist in the current file
#   and be executable.
 set xfiles = (/Net/sgil/sdc3/pkj/hetc/exec.hetc\
  /Net/sgil/sdc3/pkj/calor/exec.spect\
  /Net/sgil/sdc2/pkj/egs/exec.egsprep\
  /Net/sgil/sdc3/pkj/calor/exec.egs\
  /Net/sgil/sdc3/pkj/calor/exec.morse)
# Check that all of the efiles exist
foreach flnm ($efiles[*])
if (! -e $flnm) then
echo The file $flnm does not exist in this directory.
echo Either change to the right directory or move $flnm to this one!
echo
set runflag = "no"
endif
end

# Check that all of the xfiles exist and have their execute permission on
foreach flnm ($xfiles[*])
if (-e $flnm) then
if (! -x $flnm) then
echo The program $flnm is not executable.
echo Use the chmod u+x $flnm command and make it so!
echo
set runflag = "no"
endif
 else
echo The file $flnm does not exist in this directory.
echo Either change to the right directory or move $flnm to this one!
echo
set runflag = "no"
 endif

end

if ($runflag != "yes") then
echo CALOR will not execute until the above actions are taken!
else
  switch ($1)
    # If no arguments, run calor from the top.
    case '':
echo
echo Starting CALOR from HETC!
date '+%h %d,  %T'

        # HETC
echo
mv -f hetch.inp hetc.inp
echo   Starting exec.hetc.
date '+%h %d,  %T'
/Net/sgil/sdc2/pkj/hetc/exec.hetc
mv -f hetc.inp hetch.inp

    # Start SPECT if arg is SPECT in either upper or lower case
```

```
    case [sS][pP][eE][cC][tT]:
echo
echo   Starting exec.spect.
date '+%h %d,  %T'
/Net/sgi1/sdc3/pkj/calor/exec.spect

    # Start EGSPREP if arg is EGSP in either upper or lower case
    case [eE][gG][sS][pP]:
echo
mv -f egspreph.inp egsprep.inp
echo   Starting exec.egsprep.
date '+%h %d,  %T'
/Net/sgi1/sdc2/pkj/egs/exec.egsprep
mv -f egsprep.inp egspreph.inp
mv -f egsprep.ppr egspreph.ppr
    # Start EGS is arg is EGS in either upper or lower case
    case [eE][gG][sS]:
echo
mv -f egsh.inp egs.inp
echo   Starting exec.egs.
date '+%h %d,  %T'
/Net/sgi1/sdc3/pkj/calor/exec.egs
mv -f egs.inp egsh.inp
mv -f EGS.PPR egsh.ppr
rm -f egspreph.out

    # Start MORSE if arg is MORSE in either upper or lower case
    case [mM][oO][rR][sS][eE]:
echo
mv -f morseh.inp morse.inp
echo   Starting exec.morse.
date '+%h %d,  %T'
/Net/sgi1/sdc3/pkj/calor/exec.morse
mv -f morse.ppr morseh.ppr
mv -f morse.inp morseh.inp

    # Start EGS again if arg is MEGS in either upper or lower case
    case [mM][eE][gG][sS]:
echo
mv -f megs.inp egs.inp
echo   Starting exec.egs again.
date '+%h %d,  %T'
echo
/Net/sgi1/sdc3/pkj/calor/exec.egs
mv -f egs.inp megs.inp
mv -f EGS.PPR megs.ppr
rm -f mgams.out
        echo Starting exec.egs again.
        date '+%h %d, %T'
        echo
        mv -f egse.inp egs.inp
        echo
        /Net/sgi1/sdc3/pkj/calor/exec.egs
        mv -f egs.inp egse.inp
         echo Starting exec.analyz
        date '+%h %d, %T'
        echo
        /Net/sgi1/sdc3/pkj/calor/exec.analyz
breaksw

    # Default if arg was not recognized
    default:
goto USAGE
endsw

echo
```

```
echo CALOR completed!
date '+%h %d,  %T'

endif
goto END

USAGE:
echo 'syntax error, USAGE: calor [SPECT or EGSP or EGS or MORSE or MEGS]'
echo 'calor should be called with ONE of the above arguments or NO arguments.'

END:
```

APPENDIX E

```
#! /bin/csh -f
# This script will run the CALOR89 sequence of programs in parallel.

unset notify

# Set default of runflag to yes
set runflag = "yes"

# Check that the number of arguments used is 0 or 1
if ($#argv > 1) goto USAGE

# the following is the set of files that must exist in the current directory.
set efiles = (hetch.inp egspreph.inp egsh.inp morseh.inp megs.inp lightm.dat)

# the following lists the set of files that must exist in the current file
#   and be executable.
 set xfiles = (/Net/sgi1/sdc3/pkj/hetc/exec.hetc\
  /Net/sgi1/sdc3/pkj/calor/exec.spect\
  /Net/sgi1/sdc2/pkj/egs/exec.egsprep\
  /Net/sgi1/sdc3/pkj/calor/exec.egs\
  /Net/sgi1/sdc3/pkj/calor/exec.morse)

# Check that all of the efiles exist
foreach flnm ($efiles[*])
if (! -e $flnm) then
echo The file $flnm does not exist in this directory.
echo Either change to the right directory or move $flnm to this one!
echo
set runflag = "no"
endif
end

# Check that all of the xfiles exist and have their execute permission on
foreach flnm ($xfiles[*])
if (-e $flnm) then
if (! -x $flnm) then
echo The program $flnm is not executable.
echo Use the chmod u+x $flnm command and make it so!
echo
set runflag = "no"
endif
 else
echo The file $flnm does not exist in this directory.
echo Either change to the right directory or move $flnm to this one!
echo
set runflag = "no"
 endif

end

if ($runflag != "yes") then
echo CALOR will not execute until the above actions are taken!
else
  switch ($1)
    # If no arguments, run calor from the top.
    case '':
echo
echo Starting CALOR from HETC!
date '+%h %d,  %T'

        # HETC
echo
mv -f hetch.inp hetc.inp
echo   Starting exec.hetc.
date '+%h %d,  %T'
/Net/sgi1/sdc3/pkj/hetc/exec.hetc
```

```
mv -f hetc.inp hetch.inp

    # Start SPECT if arg is anything.
    default:
        # Make directory to run megs in.
        mkdir ./temp.megs
mv -f megs.inp ./temp.megs/egs.inp
        cp /Net/sgi1/sdc3/pkj/calor/exec.egs ./temp.megs/exec.megs

        # Make directory to run egse(electrons) in.
        mkdir ./temp.egse
mv -f egse.inp ./temp.egse/egs.inp
        cp /Net/sgi1/sdc3/pkj/calor/exec.egs ./temp.egse/exec.egse

echo
echo   Starting exec.spect.
date '+%h %d,  %T'
(/Net/sgi1/sdc3/pkj/calor/exec.spect ; echo finished exec.spect at:;\
        date '+%h %d,  %T' ) &

echo
mv -f egspreph.inp egsprep.inp
echo   Starting exec.egsprep.
date '+%h %d,  %T'
(/Net/sgi1/sdc2/pkj/egs/exec.egsprep;\
        echo exec.egsprep finished at:;\
date '+%h %d,  %T';\
mv -f egsprep.inp egspreph.inp;\
mv -f egsprep.ppr egspreph.ppr;\
echo;\
mv -f egsh.inp egs.inp;\
echo   Starting exec.egs.;\
date '+%h %d,  %T';\
/Net/sgi1/sdc3/pkj/calor/exec.egs;\
        echo exec.egs finished at:;\
date '+%h %d,  %T';\
mv -f egs.inp egsh.inp;\
mv -f EGS.PPR egsh.ppr;\
rm -f egspreph.out) &

echo
mv -f morseh.inp morse.inp
echo   Starting exec.morse.
date '+%h %d,  %T'
( /Net/sgi1/sdc3/pkj/calor/exec.morse;\
        echo exec.morse finished at:;\
date '+%h %d,  %T';\
mv -f morse.ppr morseh.ppr;\
mv -f morse.inp morseh.inp;\
echo;\

        mv -f mgams.out ./temp.megs/mgams.out;\
 echo   Starting exec.egs again.;\
date '+%h %d,  %T';\
echo;\
        cd ./temp.megs;\
./exec.megs;\
        echo megs finished at:;\
date '+%h %d,  %T';\
        cd ..;\
mv -f ./temp.megs/egs.inp ./megs.inp;\
mv -f ./temp.megs/EGS.PPR ./megs.ppr;\
        rm -f ./temp.megs/mgams.out;\
        rm -f ./temp.megs/exec.megs;\
        mv -f ./temp.megs/*.dat ./;\
        rm -rf ./temp.megs) &
```

```
echo   Starting exec.egs (for electrons) again.;\
date '+%h %d,  %T';\
echo;\
        cd ./temp.egse;\
./exec.egse;\
        echo egse finished at:;\
date '+%h %d,  %T';\
        cd ..;\
mv -f ./temp.egse/egs.inp ./egse.inp;\
mv -f ./temp.egse/EGS.PPR ./egse.ppr;\
        rm -f ./temp.egse/exec.egse;\
        mv -f ./temp.egse/*.dat ./;\
        rm -rf ./temp.egse) &

endsw

# Wait until all background processes are finished
wait
       echo starting exec.analyz
       date '+%h %d, %T'
       echo
       /Net/sgi1/sdc3/pkj/calor/exec.analyz
echo
echo CALOR completed!
date '+%h %d,  %T'

endif
goto END

USAGE:
echo 'syntax error, USAGE: pcalor [argument]'
echo 'If pcalor is called with no arguments then it runs starting with HETC.'
echo 'If ANY argument is used then parallel execution starts with SPECT, MORSE'
echo '  and EGSPREP.'

END:
```