SLAC-TN-87-3
31 July 1987
16 August 1990 (Rev.)

Producing EGS4 Shower Displays With the Unified Graphics System*

SLAC-TN--87-3-Rev.

DE90 017027

RAY F. COWAN

Laboratory for Nuclear Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

and

W. R. NELSON

Stanford Linear Accelerator Center
Stanford, California 94305

ABSTRACT

The EGS4 Code System has been coupled with the SLAC Unified Graphics System in such a manner as to provide a means for displaying showers on UGS77-supported devices. This is most easily accomplished by attaching an auxiliary subprogram package (SHOWGRAF) to existing EGS4 User Codes and making use of a graphics display or a post-processor code called EGS4PL. SHOWGRAF may be used to create shower displays directly on interactive IBM 5080 color display devices, supporting three-dimensional rotations, translations, and zoom features, and providing illustration of particle types and energies by color and/or intensity. Alternatively, SHOWGRAF may be used to record a two-dimensional projection of the shower in a device-independent graphics file. The EGS4PL post-processor may then be used to convert this file into device-dependent graphics code for any UGS77-supported device. Options exist within EGS4PL that allow for two-dimensional translations and zoom, for creating line structure to indicate particle types and energies, and for optional display of particles by type. All of this is facilitated by means of the command processor EGS4PL EXEC together with new options (5080 and PDEV) with the standard EGS4IN EXEC routine for running EGS4 interactively under VM/SP.

* Work supported by U.S. Department of Energy under contract numbers DE-AC02-76ER03069 and DE-AC03-76SF00515.

TABLE OF CONTENTS

1. EGS4 Showers and Graphic Displays

1.1. Introduction

The EGS4PL/SHOWGRAF code[1] is an add-on utility for viewing the geometrical aspects of electron-gamma showers generated by the EGS4 Code System.[2] By adding four extra subroutine calls to a non-graphics EGS4 user code, and coding one extra subroutine similar in function to HOWFAR, you get three-dimensional color graphics viewable on an IBM 5080 color graphics display or any UGS77-supported device.

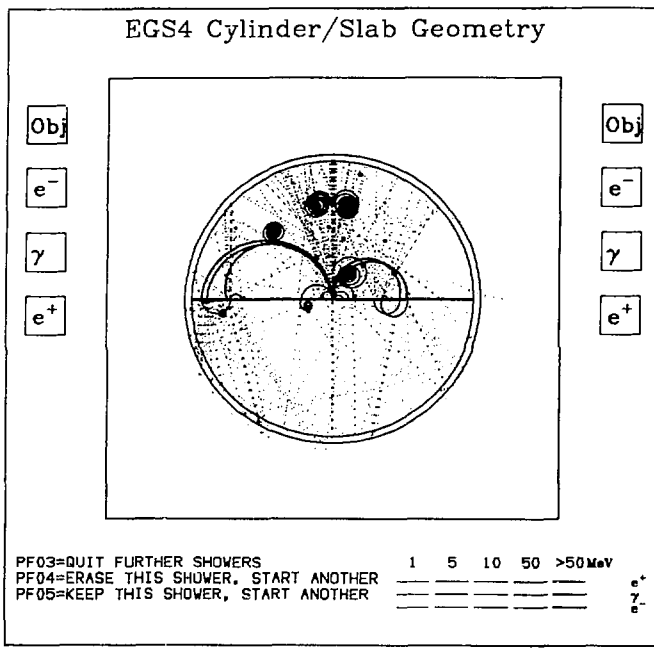

Fig. 1. Electromagnetic shower tracks in a liquid hydrogen bubble chamber. A 1 GeV photon is incident vertically upward, and traverses the chamber volume until encountering a 0.5 radiation length lead converter in the center, followed by pair production and bremstrahlung. The magnetic field is 20 kGauss. Charged tracks are shown as solid lines, photons as dotted (from user code `UPBUBBLE`).

Sample showers for a few geometries are shown in Figures 1 and 2 and also in Reference 3.

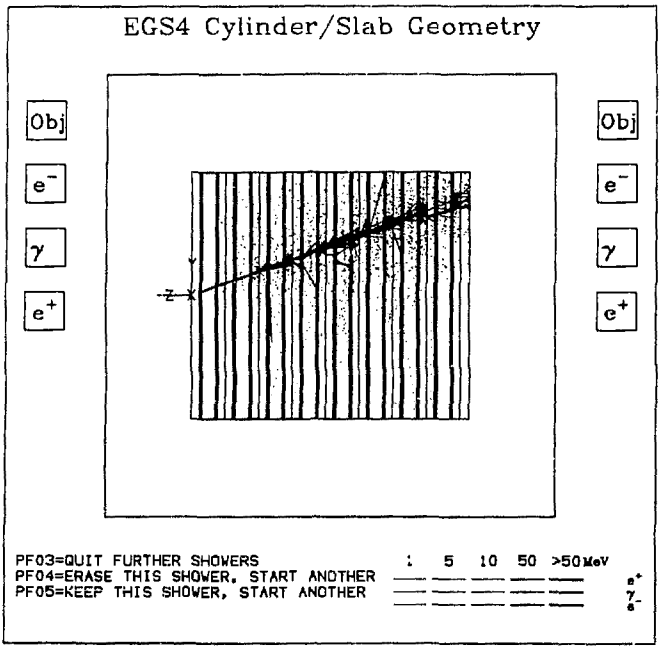

Fig. 2. Electromagnetic shower tracks in a sandwich calorimeter. A 5 GeV photon is incident horizontally from the middle of the left side onto a a sandwich calorimeter consisting of aluminum converters, wire chambers, & air separation gaps, as used in the E137 axion/photino search at SLAC[4] (from user code UPE137).

1.2. Capabilities

Particle tracks are identified by type (electron, photon, positron) and by energy. Geometry boundaries may be shown. Track type viewability may be toggled off and on, and the shower can be rotated or zoomed; multiple showers may be overplotted. After the graph is generated, the user interacts locally with the downloaded picture via the 5080 keyboard, program function (PF) keys, and analog knobs, and pick device.

Any three-dimensional color graphics display which is supported by the Unified Graphics System for Fortran 77 may be used. If you do not have access to a three-dimensional device, you can still produce two-dimensional projections of the showers for plotting on any UGS77 device. This is implemented via a graphics post-processor code (EGS4PL) which reads stored graphics information optionally produced by SHOWGRAF and does zooms, viewability toggles, *etc.* at run-time. EGS4PL cannot rotate showers; this must be done by

SHOWGRAF before the graphics information is stored.

1.3. Requirements

Code Requirements

EGS4, UGS77, and MORTRAN (as distributed with EGS4) are needed for the shower displays. EGS4 and Mortran are both on the EGS4 Distribution Tape. UGS77 is available on tape from Robert C. Beach at SLAC.[5,6] Both systems have received widespread distribution in the physics community. See your system manager to find out if you already have part, or all, of the software.

User Requirements

An understanding of EGS4 and the ability to write a non-graphics EGS4 user code are required. The best way to start is to run one of the sample "User Plot" (UP) user codes, and then add the SHOWGRAF routines to your existing user code after gaining familiarity with the system.

Helpful Additions

Familiarity with the UGS77 system is strongly recommended. Experience with two-dimensional graphics and UGS77 will be beneficial; you will already be familiar with the concepts of viewports, windows, *etc.*, as used by the Unified Graphics System.

2. How to Add SHOWGRAF to an EGS4 User Code

2.1. Where to Add SHOWGRAF Calls

Recall the basic structure of an EGS4 user code:

1. Initialization, including media and geometry (HOWFAR);
2. The repetitive shower loop (usually 1 through `NCASES`); and
3. Termination.

These can be broken down further into the standard 8-step process of the sample EGS4 user codes:

1. User override of EGS macros.
2. Pre-HATCH call initialization.
3. HATCH call.
4. HOWFAR initialization.
5. AUSGAB initialization.
6. Determination of incident particle properties.
7. SHOWER call loop.

8. Output of results.

Recall that EGS4 needs two user-supplied subroutines to function: HOWFAR and AUSGAB. The HOWFAR routine provides geometrical information that communicates to EGS4 what region a particle is in, and if the particle's next transport step will cross a region boundary. The AUSGAB routine allows accumulation of statistics for each shower, and for the sum of all showers. Each time a particle is transported EGS4 will call AUSGAB which can decide whether to accumulate any information regarding the particle (for instance, if EGS4 tells AUSGAB that it's going to discard a particle, AUSGAB may want to add the particle's energy to a histogram of energy deposited in a region).

For SHOWGRAF purposes, an additional user-written routine is required, containing geometrical constants needed by the graphics package. This routine, called HOWPL (for HOWfar-to-PLot), initializes all graphics information. It (1) defines the three-dimensional volume which contains the portions of showers and regions to plot, and (2) draws the boundaries of interest for the object being simulated. A sample HOWPL is provided with the SHOWGRAF package that uses the common cylinder/slab geometry.* This version can be used with any cylinder/slab geometry provided the common blocks `CYLDTA` and `PLADTA` contain valid coordinates. These common blocks are usually initialized in the early steps of the EGS4 user code.

The required calls to operate SHOWGRAF are shown in the following code segment (from user code `UPE137`). Note that the integer variable `IPLOT` is a flag used to control generation of graphics. If `IPLOT` = 0, then no graphics is produced—this is useful for batch mode execution of the user code; if `IPLOT` = 1, graphics is generated for the Unified Graphics System pseudo-device `PDEVLIN`;† if `IPLOT` = 2, graphics is generated for the IBM 5080 display unit. The lines added for SHOWGRAF are shown in ***SLANTED TYPE***.

```
1  DO I=1,NCASES [

                :

2     IF(IPLOT.GE.1) CALL SHOWPL(1,0);  "Initialize plotter for new shower"

3     CALL SHOWER(IQI,EI,XI,YI,ZI,UI,VI,WI,IRI,WTI);

4     IF(IPLOT.GE.1) CALL SHOWPL(3,0);  "  Terminate shower plot  "
5     NCOUNT=NCOUNT+1;
6     IXXEND=IXX;
7     ] "END OF SHOWER GENERATION LOOP"

8  TOTKE=NCOUNT*AVAILE; "TOTAL (AVAILABLE) K.E."
```

* This is a series of one or more concentric cylinders and one or more infinite planes intersecting the cylinders' axes at right angles, defining concentric rings of limited length. This description is used by the sample user codes UPCYSL and UPE137, among others.

† See Ref. 5

```
      :
9   IF(IPLOT.GE.1) CALL SHOWPL(4,0);  "  Close the graphics device  "
```

The single line:

```
IF(IPLOT.GE.1) CALL SHOWPL(2,IARG);  "  Plot the shower  "
```

is added to the routine AUSGAB. This line does most of the work: any particle which is tracked is handled via this call. SHOWPL obtains the particle's coordinates from the EGS4 stack. This works because of the strict order in which the particle stack is constructed from binary particle creation.

After adding the SHOWGRAF calls, the standard 8-step process of the sample EGS4 user codes becomes:

1. User override of EGS macros.
2. Pre-HATCH call initialization.
3. HATCH call.
4. HOWFAR initialization.
5. AUSGAB initialization.
6. Determination of incident particle properties.
7. The SHOWER loop:
   (a) SHOWPL initialization.
   (b) SHOWER call (including a new SHOWPL call in AUSGAB).
   (c) SHOWPL plot termination (clear device for a new plot).
8. Output of results.
9. SHOWPL termination (close the plotting device).

2.2. SHOWPL Arguments

The four calls are straightforward to add. The arguments to SHOWPL are indicated in the table below. The syntax is:

```
CALL SHOWPL(ISHOW,IARGD);
```

Note that IARGD is the argument passed to SHOWPL, but IARG is the argument passed to AUSGAB which may be passed on to SHOWPL.

ISHOW	IARGD	Action Taken/Requirements
1	0	Open graphics device (once) and initialize for complete new picture—*i.e.*, clear screen, perform scaling, draw object, and reset the pointers for shower track plotting/overplotting.

1	−1	Initialize for overplotting only—*i.e.*, reset the pointers for the addition of new tracks to picture. Note: `SHOWPL(1,0)` can be called as many times as necessary, but must be called at least once. Either `SHOWPL(1,0)` or `SHOWPL(1,-1)` must be called prior to calling subroutine `SHOWER` in the EGS4 User Code.
2	IARG	Plot shower tracks. Note: `SHOWPL(2,IARG)` must be called in subroutine AUSGAB of the EGS4 User Code.
3	Ignored	Pause so user can look at picture and/or utilize PF-keys: PF3 to quit plotting pictures, PF4 to draw complete new picture, or PF5 to add more tracks to same picture. Note: `SHOWPL(3,0)` must be called after `SHOWER` is called in the EGS4 User Code.
4	Ignored	Close all graphics devices. Note: `SHOWPL(4,0)` must be called before program ends.
5	Ignored	Plot a single point at the last location in color. Intended to indicate the position of created isotopes during the shower.

3. A Tour of the Routines

3.1. The HOWPL Routine

It is important to understand the information needed by the `HOWPL` routine. We will go through the cylinder/slab version of `HOWPL` section by section as an example.

This particular listing is chosen from the user code `UPE137`.

Header Information

First is the subroutine declaration and comment information.

```
1  %E
2  "************************************************************************"
3  "                                    STANFORD LINEAR ACCELERATOR CENTER"
4  SUBROUTINE HOWPL(OBJVOL,WLDVOL,VPRT3D);
5  "                                    EGS4 SUBPROGRAM -  5 JUN 1987/1500"
6  "************************************************************************"
7  "+--------------------------------------------------------------------+"
8  "|                                                                    |"
9  "|  HOWPL MORTRAN --  Required by routine SHOWPL in order to:         |"
10 "|                      A) perform object scaling                     |"
11 "|                      B) perform object drawing.                    |"
```

```
12  "|                                                                    |"
13  "|  NOTE:  This version of HOWPL incorporates the geometry of the     |"
14  "|  -----  EGS4 user code UCCYSL MORTRAN, a cylinder/slab             |"
15  "|         arrangement.  It may be used with any user code utilizing  |"
16  "|         this geometry.  HOWPL may be modified for other geometries |"
17  "|         by changing the object volume calculation and the object   |"
18  "|         drawing sections (look for them below).  HOWPL must follow |"
19  "|         SHOWPL because of Mortran3 macro definitions in the latter.|"
20  "|                                                                    |"
21  "|                                                                    |"
22  "|  Written by      R. F. Cowan                                       |"
23  "|  ----------      Laboratory for Nuclear Science                    |"
24  "|                  Massachusetts Institute of Technology             |"
25  "|                  Cambridge, MA 02139                               |"
26  "|                                                                    |"
27  "|         and      W. R. Nelson                                      |"
28  "|                  Stanford Linear Accelerator Center                |"
29  "|                  Stanford, CA 94305                                |"
30  "|                                                                    |"
31  "+--------------------------------------------------------------------+"
```

Common Blocks and Declarations

Next come the variable declarations. In particular, we pick up the common blocks containing the geometry. The common blocks CYLDTA and PLADTA contain the coordinates of the concentric cylinders and perpendicular planes that define the boundaries of the various regions.

```
32  ;COMIN/CYLDTA,PLADTA/; "  CONTAINS GEOMETRY INFO  "
```

The QPLAIN variable controls drawing user-interface information on the plot. This info (such as pick-device selectable indicators, energy scale legend, *etc.*) are most useful when the shower is viewed interactively on the IBM 5080 screen, and not when two-dimensional hardcopy is being made. When set true, this switch removes the extra legends from the plot.

```
33  COMMON/PLAIN/QPLAIN;          "  T=plain picture (default in BLOCK DATA "
34                                "  F=5080 borders/titles/icons "
35  LOGICAL*1 QPLAIN;
```

The next common block, SHWRSC, contains scale factors. The axis scales are done independently so that the object and shower may be plotted with different scales; this allows, for instance, convenient display of objects that are "long and thin" (like a beampipe) or "short and wide" (like a pancake) by stretching or compressing the display of one axis with respect to the others. The overall scale factor variable SCFACT is used to expand or compress the entire object uniformly within the *world volume** for controlling effects of clipping planes and translations.

* See Ref 5.

By default, XSCALE, YSCALE, and ZSCALE are set to unity; SCFACT is set to 2.5 in a BLOCK DATA section.

```
36  COMMON/SHWRSC/XSCALE,YSCALE,ZSCALE,SCFACT;  "  Mult. scale factors"
```

Next are local variables used to draw the object, in this case a series of concentric cylinders truncated with perpendicular planes. The parameter $NCIRCLE sets the number of straight line segments to use around a circle perimeter when displaying it. The more line segments, the better the circle looks; but also the more memory and time required. This is important because lots of circles will rapidly fill the 5080's memory.

```
37  REAL*4 CYLRAD($MXCYLS);  "  Cylinder radii  "

38  PARAMETER $NCIRCLE=51; " Number of segments to use in drawing circles"
39  REAL*4 XCYL($NCIRCLE),YCYL($NCIRCLE),ZCYL($NCIRCLE); " Used in object "
```

Common block PASSIT contains more information on the geometry: the particle discard flag for each region, and the number of cylinders and planes.

```
40  COMMON/PASSIT/IRDISC($MXREG),NR,NZ; "NR=number of cylinders, NZ=   "
41                                      " number of slabs "
```

In order to initialize the Unified Graphics System, we must decide whether the graphics device has interactive capabilities. QINTER and INTACT record this.

```
42  LOGICAL*1 QINTER;  "  T=we are using an interactive plotting device  "
43  INTEGER*4 INTACT(10);
```

Now come the fundamental Unified Graphics System parameters that define how the three-dimensional display will appear when sent to the device. These parameters define the three-dimensional *world volume*, *viewport*, *object volume*, *viewing direction*, and *type of projection*. See the initialization code below and Reference 5 for more details.

```
44  REAL*4 WLDVOL(3,2); " 3D world volume "
45  REAL*4 OBJVOL(3,2); " 3D object volume  "
46  REAL*4 VPRT3D(2,2); " 3D viewport  "
47  REAL*4 EYEPT(3); "Eye point for the parallel/perspective transformation"
48  REAL*4 UPDIR(3)/0.0,1.0,0.0/;  "  Up direction on screen  "
49  REAL*4 PFLAG/0.00/;  "  Projection flag:  0 --> parallel projection,  " /
50                       "                    >0 --> perspective projection "
```

The three variables XAXIS, YAXIS, and ZAXIS contain the coordinates in three-dimensions for drawing the origin of the coordinate system. These produce a right-handed coordinate system on the screen.

The *blanking bits* variable, AXESBB, contains one bit for every pen stroke used in drawing the axes; a '1' means pen down, a '0' means pen up; the high order bit is used first.

```
51  "  3D axes coordinates and blanking bits  "
```

```
52  REAL*4      XAXIS(5)/5*0.0/,
53              YAXIS(5)/5*0.0/,
54              ZAXIS(5)/5*0.0/;
55  INTEGER*4   AXESBB/ZB0000000/;  "  Blanking bits  "
```

To allow for control of viewability of different parts of the object, shower, *etc.*, we use several variables, '`IVWxxx`', which indicate whether or not the particular part of the display is viewable. A value $\neq 0$ indicates the part is drawn.

```
56  "  Graphic segment identifiers and view flags  "

57  COMMON/IVIEW/IVWOBJ,IVWELC,IVWPHT,IVWPOS,IDGOBJ,IDGELC,IDGPHT,IDGPOS;
```

The *graphic segments* contain the Unified Graphics System-processed information ready for transmission to the graphics device. Remember that when a segment fills up, it must be transmitted to the device (via a '`CALL UGWRIT`') before any more information can be stored. The '`$xxxSIZE`' parameters define the size of the graphics segments.

```
58  "  The graphic segments  "

59  COMMON/GRAPHS/ISEG,SEGMNT($SEGSIZE),OBJECT($OBJSIZE),ELCTRN($TRKSIZE),
60                PHOTN($TRKSIZE),POSTRN($TRKSIZE);
```

Object Volume Calculation

Now the executable code starts.

First the maximum extent of the object is calculated using the units of the EGS4 coordinate system (centimeters, or inches, or whatever). The goal is to find the maximum dimensions to be displayed in each of the coordinate directions x, y, and z.

For this simple cylindrical geometry, these are the maximum cylinder radii and the distance from the first plane to the last plane.

These values are stored in array `OBJVOL` as required by the Unified Graphics System.

```
61  "  Calculate 3D world and object volumes and 3D viewport  "

62  "  First the object volume (in whatever units EGS4 is using)  "

63  "  ***** This calculation of the object volume (OBJVOL) depends *****  "
64  "  ***** on the use of the EGS4 cylinder/slab geometry in the   *****  "
65  "  ***** UCCYSL user code or a similar user code and must be    *****  "
66  "  ***** modified if a different geometry is used.              *****  "

67  NRADII=NR-1;  "  Number of radii  "
68  DO IRAD=1,NRADII [
69    CYLRAD(IRAD)=SQRT(CYRAD2(IRAD));  "  Cylinder radii  "
```

```
70    ]

71  RADMAX=CYLRAD(NRADII);  "  Cylinder maximum radius  "
72  XLO=-RADMAX*XSCALE;
73  YLO=-RADMAX*YSCALE;
74  ZLO=PCOORD(3,1)*ZSCALE;
75  XHI=RADMAX*XSCALE;
76  YHI=RADMAX*YSCALE;
77  ZHI=PCOORD(3,NZ)*ZSCALE;
78  OBJVOL(1,1)=XLO;
79  OBJVOL(2,1)=YLO;
80  OBJVOL(3,1)=ZLO;
81  OBJVOL(1,2)=XHI;
82  OBJVOL(2,2)=YHI;
83  OBJVOL(3,2)=ZHI;

84  "  ***** End of object volume (OBJVOL) calculation.              *****  "
```

World Volume Calculation

Next the *world volume* is calculated in which the Unified Graphics System will display the object and shower. Anything with coordinates outside the world volume will be omitted from the display.

This calculation uses the largest ($high - low$) dimension found in the object volume calculation for the length of all three sides of the world volume. This is done because the object may be rotated inside the world volume, and we must allow room so the object is not truncated during rotations (although this can still happen by various combinations of zoom, clipping plane motion, and object/shower translations).

The center of the world volume is calculated to coincide with the center of the object volume.

An initial overall scale factor, SCFACT, is applied to shrink or expand the initial display of the object with the world volume. The default value is unity. Note that this acts on all three axes equally; changing the aspect ratio of one axis with respect to another is accomplished by the variables XSCALE, YSCALE, and ZSCALE, which have already been used in the object volume calculation.

No changes are normally needed in the world volume calculation since it is based on the object volume calculation above.

First find the maximum length of the object.

```
85  "-------------------------------------------------------------------"
86  "-------- User should not change the following section of code --------"
87  "-------------------------------------------------------------------"

88  "  Make the world volume bigger than the object volume  "
```

```
89  VAL=0.0;
90  DO IC=1,3 [
91    VAL=AMAX1(VAL,OBJVOL(IC,2)-OBJVOL(IC,1));
92    ]
```

Center the world volume on the center of the object volume.

```
93  DO IC=1,3 [
94    CTR=0.5*(OBJVOL(IC,1) + OBJVOL(IC,2));
95    WLDVOL(IC,1)=CTR - 0.5*VAL*SCFACT;
96    WLDVOL(IC,2)=CTR + 0.5*VAL*SCFACT;
97  ]
```

Three-dimensional Transformation

There are more parameters to calculate for the 3D-to-2D projection transformation. Imagine this as a viewer looking at the rectangular world volume and a flat screen behind it. The projection determines the 2D-coordinates of each 3D point by projecting a straight line from the viewer's position (the *eyepoint*) through the 3D point and onto the flat screen. We must specify the distance from the viewer to the world volume, the orientation of the eyepoint with respect to the world volume (the "direction from which we are looking"), and the rotation of the flat screen about the projection direction ("Which way is up?" on the 2D-display).

Two eyepoint selections are listed here, one of which should be commented out in the Mortran source. The first positions the eyepoint at the high end of the world volume on the z axis. The second is a "sideways" view: the eyepoint is positioned at the "low-x, mid-y, mid-z" position on the world volume boundary.

Note that the eyepoint *must never lie within the world volume*, although (as here) it may lie on its boundary. This is sufficient for *parallel* projections where the eyepoint is considered to be infinitely far away. This should be changed if perspective projections are needed.

```
98  "  Calculate the eyepoint  "
99  "The first is for end-on and the second is for sideways view"
100 "EYEPT(1)=0.0; EYEPT(2)=0.0; EYEPT(3)=WLDVOL(3,2);"
101 EYEPT(1)=WLDVOL(1,1); EYEPT(2)=0.0;
102 EYEPT(3)=(WLDVOL(3,1) + WLDVOL(3,2))/2;
```

Now call the Unified Graphics System routines to define (1) the world volume in terms of a three-dimensional viewport (defined in the BLOCK DATA section), and (2) the transformation.

```
103 CALL UG3WRD('PUT',VPRT3D,WLDVOL);
104 CALL UG3TRN('PUT',OBJVOL,EYEPT,UPDIR,PFLAG);
```

Check the interactive status of the selected graphics device. If it allows user interaction, enable the devices accordingly.

```
105 CALL UGINFO('ILEVEL',JUNK,INTACT);  "  Is it interactive?   "
```

```
106                                          "  1=non-interactive,   "
107                                          "  2=slave display,     "
108                                          "  3=fully interactive  "
109  QINTER=.NOT.(INTACT(1).EQ.1 .OR. INTACT(1).EQ.2);
110  IF(QINTER) [
111    CALL UGENAB('KEYBOARD,PICK,BUTTON,STROKE');" Enable local input from "
112    ]                                          " the 5080 display unit   "
```

Drawing the Object

This section constructs an outline of the principle features of the object as a reference for examining the shower. There is much room for personal creativity at this point. Typical practice has been to draw a wire frame version of object showing the boundaries (in this case) of cylinders and planes. The user may draw as much or as little of the object as he finds necessary to interpret the position of the particle tracks in the shower.

The user *must* be familiar with the Unified Graphics System routines UG3PLN, UG3LIN, and UG3TXT to draw the object, axes, and axis labels.[6] The user must also understand the operation of the CHKBUF routine which outputs graphic segments to the device as they become full.

First the circular intersections of the cylinders with the planes are drawn by filling the arrays XCYL, YCYL, ZCYL with line segment endpoints which approximate the circles. Then, after ensuring enough space in the graphic segment, these circle approximations are transmitted to the device.

```
113  "-------------------------------------------------------------------"

114  "  ***** This construction of the object outlines depends       *****  "
115  "  ***** on the use of the EGS4 cylinder/slab geometry in the   *****  "
116  "  ***** UCCYSL user code or a similar user code and must be    *****  "
117  "  ***** modified if a different geometry is used.              *****  "

118  "  Draw the outline of the cylinder/slab geometry  "

119  ISEG=2;  "  Set UGXERR segment pointer to the object segment while  "
120           "  we are drawing it so UGXERR can clear it every time we  "
121           "  manage to fill  it up  "

122  TWOPI=8.0*ATAN(1.0);
123  SEGNUM=FLOAT($NCIRCLE-1);
124  DO ICYL=1,NRADII [  "  Draw each cylindrical ring  "
125    DO IPLN=1,NZ  [  "  In each plane  "
126      DO IANG=1,$NCIRCLE [  "  if > 200 points makes a decent circle  "
127        ANGLE=TWOPI*FLOAT(IANG-1)/SEGNUM;
128        XCYL(IANG)=CYLRAD(ICYL)*COS(ANGLE)*XSCALE;
129        YCYL(IANG)=CYLRAD(ICYL)*SIN(ANGLE)*YSCALE;
130        ZCYL(IANG)=PCOORD(3,IPLN)*ZSCALE;
```

```
131         ]
132     CALL CHKBUF(OBJECT,IDGOBJ,$OBJSIZE);  "  Make room in the segment  "
133     CALL UG3PLN('MAGENTA,BRIGHT',XCYL,YCYL,ZCYL,$NCIRCLE,1,1,
134                 OBJECT);
135       ]
```

Now draw the outer radius of each cylinder as a line parallel to the cylinder axis in the y-z plane.

```
136     DO IPLN=2,NZ [
137       CALL CHKBUF(OBJECT,IDGOBJ,$OBJSIZE);  "  Make room in the segment  "
138       XPLN1=0.0; XPLN2=XPLN1;
139       YPLN1=CYLRAD(ICYL)*YSCALE; YPLN2=YPLN1;
140       ZPLN1=PCOORD(3,IPLN-1)*ZSCALE; ZPLN2=PCOORD(3,IPLN)*ZSCALE;
141       CALL UG3LIN('MAGENTA,BRIGHT',XPLN1,YPLN1,ZPLN1,0,OBJECT);
142       CALL UG3LIN('MAGENTA,BRIGHT',XPLN2,YPLN2,ZPLN2,1,OBJECT);
143       YPLN1=-YPLN1; YPLN2=-YPLN2;
144       CALL UG3LIN('MAGENTA,BRIGHT',XPLN1,YPLN1,ZPLN1,0,OBJECT);
145       CALL UG3LIN('MAGENTA,BRIGHT',XPLN2,YPLN2,ZPLN2,1,OBJECT);

146       ]

147     ]
```

Coordinate Axes

Draw the indicative coordinate axes. These are set to be 1/10 of the length of the object in each axis direction, and are centered on the point $(0,0,0)$.

```
148   IF(.NOT.QPLAIN) [ "3D axes section"
149   "  Axis length is 1/10 of maximum object dimension "
150     AXISLN=AMAX1(   1.0,0.1*(OBJVOL(1,2)-OBJVOL(1,1)));
151     AXISLN=AMAX1(AXISLN,0.1*(OBJVOL(2,2)-OBJVOL(2,1)));
152     AXISLN=AMAX1(AXISLN,0.1*(OBJVOL(3,2)-OBJVOL(3,1)));

153     XAXIS(2)=AXISLN;
154     YAXIS(3)=AXISLN;
155     ZAXIS(5)=-AXISLN;

156   " Draw the axes "
157     CALL UG3PLN('CYAN ,BRIGHT',XAXIS,YAXIS,ZAXIS,5,AXESBB,-5,OBJECT);

158     AXL=1.15*AXISLN;  "  Text position slightly beyond end of axis  "

159     " Draw axes labels "
160     CALL UG3TXT('WHITE,BRIGHT',AXL,0.0,0.0,'X',OBJECT);
161     CALL UG3TXT('WHITE,BRIGHT',0.0,AXL,0.0,'Y',OBJECT);
162     CALL UG3TXT('WHITE,BRIGHT',0.0,0.0,-AXL,'-Z',OBJECT);
163     ]
```

Cleanup

We are almost done. We transmit any remaining data in the graphic segment and record its identification, IDGOBJ, for later use by the viewability toggle routines (see routine CHKBUF).

```
164  IDGOBJ=IDGOBJ+1;  "  Increment object segment id  "
165  CALL UGWRIT(' ',IDGOBJ,OBJECT);  " Output remaining parts of object "
166  CALL UGINIT('CLEAR',OBJECT,$OBJSIZE);  "  and clear the object segment "

167  "  ***** End of object drawing code.                                  *****  "

168  RETURN;
169  END;
```

This is the end of the object drawing code.

Block Data Section

This code section defines the default values for most of the variables controlling the graphics display. First come the variable declarations.

```
1   %E
2   "*************************************************************************"
3   "                                  STANFORD LINEAR ACCELERATOR CENTER"
4   BLOCK DATA;
5   "                                  EGS4 SUBPROGRAM -  8 JUN 1987/1500"
6   "*************************************************************************"
7   "   For SHOWPL-HOWPL                                                      "

8   COMMON/SHWRSC/XSCALE,YSCALE,ZSCALE,SCFACT;  "  Mult. scale factors"

9   COMMON/SHOWRP/ENRGCT(6),  " Variables the user can change to "
10                XTOPLN,     " configure the plot to his needs  "
11                YTOPLN,
12                COLORS(3),TOPCON,TOPTXT,TOPCAS;

13  COMMON/PLAIN/QPLAIN;         "  T=plain picture (default set below) "
14                               "  F=5080 borders/titles/icons "
15  LOGICAL*1 QPLAIN;

16  CHARACTER*8 COLORS;
17  CHARACTER*50 TOPCON,TOPTXT,TOPCAS;
```

The QPLAIN flag omits the interactive-mode info if set to be true. This is useful for making hardcopy-only plots (via the EGS4PL post-processor).

```
18  DATA QPLAIN/.TRUE./;  " Default is for plain picture "
```

These scaling values control the relative aspect ratios of the three axes with respect to each other and SCFACT controls the initial zoom factor of the object/shower with the world volume.

```
19 DATA XSCALE/1.0/,  "  Default is unity scaling  "
20      YSCALE/1.0/,
21      ZSCALE/1.0/;

22 DATA SCFACT/2.5/;  "Scale factor between world and object volumes"
23                    " (affects zooming capability)"
```

The values in the ENRGCT ("ENeRGy CuT") array set the energy boundaries in MeV for the intensity cueing. Higher-energy tracks are plotted with a brighter intensity.

```
24 DATA ENRGCT/0.0,1.0,5.0,10.0,50.0,1.0E8/; " Brightness control  "
25 "  If E(NP) < ENRGCT(1), particle is not plotted; if E(N) > ENRGCT(1), "
26 "  plot it as VDIM, etc.  If E(NP) > ENRGCT(6), do not plot it.  "
```

Select colors for electron, photon, positron tracks respectively.

```
27 DATA COLORS/'   GREEN',   "  Electron color  "
28             ' YELLOW',   "  Photon color    "
29             '    RED'/;  "  Positron color  "
```

Titles for the interactive plot are next: TOPTXT is the top title on the display, TOPCAS is the case-control information for the Unified Graphics System, XTOPLN and YTOPLN are the coordinates for positioning the top line text, and TOPCON is a control string telling the Unified Graphics System how to position it with respect to the coordinates, and what color to use, and what size characters to use (in 1/10 inch increments).

```
30 "  Top line text  "
31 DATA TOPTXT  /'EGS4 CYLINDER/SLAB GEOMETRY                    '/;
32 "  Top line case control  "
33 DATA TOPCAS  /'      LLLLLL  LLL  LLLLLL                      '/;
34 "  Coordinates of top line text  "
35 DATA XTOPLN/50.0/,YTOPLN/97.0/;
36 "  Top line characteristics  "
37 DATA TOPCON  /'CENTER,SIZE=3.0,CYAN                           '/;

38 END;
```

This is the end of the BLOCK DATA section.

4. Generating Hardcopy: the EGS4PL Post-processor

4.1. Description

The color display with rotational capability is enjoyable to look at, but hardcopy is necessary from time to time. Three possibilities exist: (1) color hardcopy via hardware attached to your display head; (2) color photography; and (3) non-color hardcopy on a two-dimensional plotting device (such as a laser printer).

Option (1) is installation-specific: either you have a hardcopy screen-dump device, which "takes a picture" of the current screen and prints it, or you don't. Some devices may be capable of producing color transparencies as well as color paper copy.

Option (2), photography, generally provides the best reproduction of the screen and is quite inexpensive. A dark room, a solid tripod, a 35 mm SLR camera, and a good color slide film produce excellent results after a little experimentation with exposure times and f/stops. Additionally, many photo labs will make full-size transparencies from color slides. The authors have made many excellent slides in this manner for use in presentations.

Option (3) is an alternative when simple black and white hardcopy is sufficient. In this case the user code generates an intermediate graphics file which contains the same information that would otherwise be plotted on the screen. A post-processor EGS4PL reads in this information, and generates two-dimensional hardcopy for any UGS-supported plotting device.

There are limitations inherent to this scheme: *no three-dimensional rotations are available at post-processor time*, although the 3D-to-2D transformation can be set to give any rotation when the intermediate file is created. Also, some compromise in designating particle types and energies must be made since color is not available.

Subject to these limitations, the EGS4PL processor does have a fairly comprehensive set of operations it can perform on the plot. These are:

1. Selection of output device. Usually a laser printer or versatec, but could be any UGS77 device.

2. Selective plotting of particle types. This emulates the viewability toggle of the pick device on the 5080.

3. Three zoom capabilities: zoom in x (the horizontal direction of the hardcopy), zoom in y (the vertical direction of the hardcopy), and simultaneous zoom in both directions (similar to the zoom knob on the 5080). Default zoom values are unity.

4. Translations in x and y. Default values are zero, which means that the center of the 2D viewport is placed at the center of the hardcopy viewport. Values are specified in units of screen halfwidth or halfheight; a value of '-1.0' shifts the picture half a screen left or down; a value of '$+1.0$' shifts it half a screen right or up.

5. Drawing an enclosing box. Without the 5080's display boundary, some difficulty may be encountered in determining where particle tracking ends because of region cuts in EGS4 versus scissoring at the boundary of the screen. This options draws a rectangular box around the border of the screen to mark the viewport area.

4.2. EGS4PL Examples

Arguments and Options

The EGS4PL processor is invoked via the EGS4PL executive command file. The syntax is:

EGS4PL *pseudo-device_filespec* (*output_device particle_type_selection box_selection translations zooms*

Zero or more options may be placed after the '(' and are allowed in any order. Arguments and options are chosen from the following list:

pseudo-device_filespec — A required filename and optional filetype and filemode of the 2D graphics file. The default filetype is 'PDEVLIN' and default filemode is '*'.

output_device — One of:

SEQ4010 – Tektronix sequential plot file.
IMPRT10 – Imagen imPRESS language @ 240 DPI.
IMGN300 – Imagen imPRESS language @ 300 DPI.
TALARIS – Talaris laser printer.
VEP12FF – Versatec plot file at 200 DPI.

particle_type_selection — This removes tracks of a selected charge from the plot:

IQOFF *charge charge charge*

box_selection — When selected, causes a box to be drawn around the plot boundary:

BOX

translations — These specify independent horizontal and vertical shifts of the picture in units of halfscreen dimensions. Defaults are zero, and typical values are between −1 and +1:

XTRANS *value* YTRANS *value*

zooms — These specify independent horizontal and vertical zooms and a global zoom (defaults are unity):

XZOOM *value* YZOOM *value* ZOOM *value*

The processor will compile, load, and run the code, creating an output file which can be transmitted to the output device.

Generating the Pseudo-graphics file

Before running EGS4PL, the pseudo-graphics file must be created. On VM/SP systems, this is accomplished by specifying 'PDEV' as the first argument to the EGS4IN exec. On VAX/VMS, the standard command file **EGS4.COM** (file #51 on the EGS4 Distribution Tape) must have two extra modifications* made: change lines 49–56 from

```
$LINKU:                          !USER MAY INCLUDE SOME MORE MODULES HERE
$LINK/EXECUTABLE='P1'/MAP/FULL 'P1',EGS'P1'.OBJ,[DAVE4.EGS4]CPUTIME, -
  LA120,DRPLT
$GOTO EXECUTE
$!
$LINKA:
$LINK/EXECUTABLE='P1'/MAP/FULL 'P1',[DAVE4.EGS4]CPUTIME,LA120,DRPLT
$!
```

to read

```
$LINKU:                          !USER MAY INCLUDE SOME MORE MODULES HERE
$LINK/EXECUTABLE='P1'/MAP/FULL 'P1',EGS'P1'.OBJ,[DAVE4.EGS4]CPUTIME, -
  LA120,DRPLT, -
  DISK1:[UGSYS]NUCEUS+SIMPLEX+DUPLEX+PDEVLIN+PDEVUGSX, -
  DISK1:[UGSYS]OBJLIB/LIB  !  EGS4PL MOD
$GOTO EXECUTE
$!
$LINKA:
$LINK/EXECUTABLE='P1'/MAP/FULL 'P1',[DAVE4.EGS4]CPUTIME,LA120,DRPLT, -
  DISK1:[UGSYS]NUCEUS+SIMPLEX+DUPLEX+PDEVLIN+PDEVUGSX, -
  DISK1:[UGSYS]OBJLIB/LIB  !  EGS4PL MOD
$!
```

Also change lines 66–69 from

```
$IF(P4.NES."") THEN -                                  !FROM PEGS4
   ASSIGN [DAVE4.PEGS4.DAT]'P4'.DAT     FOR012
$RUN 'P1'
$DEASSIGN/ALL
```

to read

```
$IF(P4.NES."") THEN -                                  !FROM PEGS4
   ASSIGN [DAVE4.PEGS4.DAT]'P4'.DAT     FOR012
$ASSIGN 'P1'.PDEVLIN FOR099  !  EGS4PL MOD
$RUN 'P1'
$DEASSIGN/ALL
```

* In addition to the normal changes in directory specifications

IBM VM/SP Systems

Here are a few examples. It is assumed that the user code UPBUBBLE has been executed with the 'PDEVLIN' option selected during job creation (*i.e.*, the job was run by saying 'EGS4IN PDEV', not 'EGS4IN 5080' or simply 'EGS4IN'), and a pseudo-graphics file UPBUBBLE PDEVLIN exists.

1. Plot the picture using defaults:

```
EGS4PL UPBUBBLE
```

2. Plot only charged tracks and draw a box:

```
EGS4PL UPBUBBLE ( IQOFF 0 BOX
```

3. Shift the plot to the left, and zoom in globally to look at some particular feature in detail:

```
EGS4PL UPBUBBLE ( XTRANS -.3 ZOOM 2.5
```

4. Plot photons only, draw a box, and zoom horizontally:

```
EGS4PL UPBUBBLE ( IQOFF -1 1 BOX XZOOM 2
```

5. Plot photons only, draw a box, and generate the output for a sequential Tektronix device:

```
EGS4PL UPBUBBLE ( IQOFF -1 1 BOX SEQ4010
```

DEC VAX/VMS Systems

Here are VAX/VMS versions of the options listed above. Note that on this machine, EGS4PL is installed as a foreign command using the specifications in the command language definition file EGS4PL.CLD; this allows convenient specification of options in standard VAX/VMS '/OPTION=*value*' format.

1. Plot the picture using defaults:

```
$ EGS4PL UPBUBBLE
```

2. Plot only charged tracks and draw a box:

```
$ EGS4PL UPBUBBLE/IQOFF=0/BOX
```

3. Shift the plot to the left, and zoom in globally to look at some particular feature in detail:

```
$ EGS4PL UPBUBBLE/XTRANS=-.3/ZOOM=2.5
```

4. Plot photons only, draw a box, and zoom horizontally:

```
$ EGS4PL UPBUBBLE/IQOFF=(-1,1)/BOX/XZOOM=2
```

5. Plot photons only, draw a box, and generate the output for a sequential Tektronix device:

```
$ EGS4PL UPBUBBLE/IQOFF=(-1,1)/BOX/SEQ4010
```

Appendix A

SHOWGRAF Code Listing

A.1. The Standard SHOWGRAF Suite

The standard version of SHOWGRAF, written in Mortran, is included here. This listing includes the SHOWPL main routine and its required utility routines UGXERR, CHKBUF, OMTSEG, and UGXLIN. Also included is a version of the user-written routine HOWPL specific to the common cylinder/slab geometry. See Chapter 3 for a guide to HOWPL.

This is the same version as supplied with the EGS4 distribution.

Code Listing of SHOWGRAF.MORTRAN

```
%E
"+-------------------------------------------------------------------------+"
"|                                                                         |"
"|                          S H O W G R A F                                |"
"|                                                                         |"
"|        SUBROUTINE PACKAGE FOR CREATING EGS4 SHOWER PICTURES             |"
"|                                                                         |"
"+-------------------------------------------------------------------------+"
"|              to be used with existing EGS4 User Codes                   |"
"+-------------------------------------------------------------------------+"

"*************************************************************************"
"                                    STANFORD LINEAR ACCELERATOR CENTER"
SUBROUTINE SHOWPL(ISHOW,IARGD);
"                                    EGS4 SUBPROGRAM - 19 JAN 1989/1530"
"*************************************************************************"
"+-------------------------------------------------------------------------+"
"|                                                                         |"
"|  SHOWPL MORTRAN -- Makes plots of EGS4 showers on 2D or 3D graphic      |"
"|                    devices using the Unified Graphics System for        |"
"|                    FORTRAN 77.                                          |"
"|                                                                         |"
"|                                                                         |"
"|  NOTE:  Subroutine SHOWPL may be used with any EGS4 User Code in        |"
"|  -----  order to plot the vectors associated with the particles in      |"
"|         the shower.  Subroutine HOWPL is required by SHOWPL in          |"
"|         order to establish scaling as well as to draw the object        |"
"|         volume (as defined by subroutine HOWFAR in the User Code).      |"
"|                                                                         |"
"|                                                                         |"
"|  Written by     R. F. Cowan                                             |"
"|  ----------     Laboratory for Nuclear Science                          |"
"|                 Massachusetts Institute of Technology                   |"
"|                 Cambridge, MA 02139                                     |"
"|                                                                         |"
"|         and     W. R. Nelson                                            |"
```

```
"|                Stanford Linear Accelerator Center                  |"
"|                Stanford, CA 94305                                  |"
"|                                                                    |"
"|--------------------------------------------------------------------|"
"|                        GRAPHICS DEVICES                            |"
"|--------------------------------------------------------------------|"
"|  Supported graphic devices          UGS77 Device Name  IDEV-value  |"
"|  --------------------------         -----------------  ----------  |"
"|                                                                    |"
"|  Graphics pseudo-device                   PDEVLIN           1      |"
"|  (produces an output file suitable for plotting on several dif-    |"
"|   ferent two-dimensional devices via the program EGS4PL.)          |"
"|                                                                    |"
"|  IBM 5080 Graphics System.  Inter-        IBM5080           2      |"
"|  active, 3D, color.                                                |"
"|                                                                    |"
"|  Note:   IDEV is set with a DATA statement in the initialization   |"
"|          section of SHOWPL.  IDEV can also be re-set with IPLOT.   |"
"|--------------------------------------------------------------------|"
"|                                                                    |"
"|  Revision History:                                                 |"
"|  -----------------                                                 |"
"|                                                                    |"
"|  861116 RFC,WRN  CREATION DATE.                                    |"
"|  861118 RFC,WRN  HOWPL-SHOWPL separation.                          |"
"|  861121 RFC,WRN  Cleanup of bugs, creation of intensity legend.    |"
"|  861126 RFC,WRN  Square world volume, scaling, misc.               |"
"|  870107 RFC,WRN  Added IARGD=-1 option to SHOWPL(1,IARGD).         |"
"|  870531 RFC,WRN  Added IDEV option to select 2D or 3D devices.     |"
"|  880506 RFC,WRN  Added use of either standard or extended PF key   |"
"|                  reporting format.                                 |"
"|                  Added ISHOW=5 case to draw isotope positions      |"
"|                  in color.                                         |"
"|  880831 RFC,WRN  Fixed UGXERR to avoid conflict with HANDYPAK.     |"
"|  890118 RFC,WRN  Changed IDEV definitions from 1 (PDEVLIN) and     |"
"|                  0 (IBM5080) to 1 (PDEVLIN) and 2 (IBM5080).       |"
"|                  Added IPLOT variable in COMMON/SHOWGF/ to control |"
"|                  IDEV from EGS4 User Code.                         |"
"|  890119 RFC,WRN  Bug (typo) fixed in energy legend (e+,e- reversed)|"
"|                  Also changed energies in legend from total to     |"
"|                  kinetic energy.                                   |"
"|                                                                    |"
"|  For a full description of the 'Unified Graphics System For        |"
"|  FORTRAN 77' routines used here see the 'UGS77 Programming         |"
"|  Manual' by Robert C. Beach, CGTM Number 203, rev. November 1985,  |"
"|  Stanford Linear Accelerator Center, Stanford, CA USA.             |"
"|                                                                    |"
"+--------------------------------------------------------------------+"
;  "---------- BUFFER FLUSH SEMICOLON ----------"
"+--------------------------------------------------------------------+"
"|                                                                    |"
"|   ISHOW  IARGD          Action Taken/Requirements                  |"
"|   -----  -----          -------------------------                  |"
"|                                                                    |"
```

```
90  "|     1       0     Open graphics device (once) and initialize for      |"
91  "|                   complete new picture---i.e., clear screen, perform  |"
92  "|                   scaling, draw object, and reset the pointers for    |"
93  "|                   shower track plotting/overplotting.                 |"
94  "|                                                                       |"
95  "|           -1      Initialize for overplotting only---i.e., reset the  |"
96  "|                   pointers for the addition of new tracks to picture. |"
97  "|                                                                       |"
98  "|                   Note: SHOWPL(1,0) can be called as many times as    |"
99  "|                         necessary, but must be called at least once.  |"
100 "|                         Either SHOWPL(1,0) or SHOWPL(1,-1) must be    |"
101 "|                         called prior to calling subroutine SHOWER in  |"
102 "|                         the EGS4 User Code.                           |"
103 "|                                                                       |"
104 "|     2    IARG     Plot shower tracks.  Note: SHOWPL(2,IARG) must be   |"
105 "|                   called in subroutine AUSGAB of the EGS4 User Code.  |"
106 "|                                                                       |"
107 "|     3   Dummy     Pause so user can look at picture and/or utilize    |"
108 "|                   PF-keys:  PF3 to quit plotting pictures,            |"
109 "|                             PF4 to draw complete new picture,         |"
110 "|                             PF5 to add more tracks to same picture.   |"
111 "|                                                                       |"
112 "|                   Note: SHOWPL(3,0) must be called after SHOWER is    |"
113 "|                         called in the EGS4 User Code.                 |"
114 "|                                                                       |"
115 "|     4   Dummy     Close all graphics devices.  Note: SHOWPL(4,0)      |"
116 "|                   must be called before program ends.                 |"
117 "|                                                                       |"
118 "|     5   Dummy     Plot a single point at the last location in color.  |"
119 "|                   Intended to indicate the position of created        |"
120 "|                   isotopes during the shower.                         |"
121 "|                                                                       |"
122 "+-----------------------------------------------------------------------+"
123 ;  "---------- BUFFER FLUSH SEMICOLON ----------"

124 "  The buffer size parameter--does not normally need  to be changed  "
125 PARAMETER $TRKSIZE=2000;  "  (it's determined by 5080 hardware)  "
126 PARAMETER $OBJSIZE=5000; "  Size of object graphic segment  "
127 PARAMETER $SEGSIZE=5000;  "  Size of general graphic segment  "

128 LOGICAL*1 QDBGPL/.FALSE./;  "  Debugging flag  "

129 ;COMIN/CYLDTA,PLADTA/; "Contains geometry information"
130 COMIN/EPCONT,STACK/;   "Contains particle information"
131 COMIN/USEFUL/;         "Contains the parameter PRM"

132 REAL*4 CYLRAD($MXCYLS);  "  Cylinder radii  "

133 PARAMETER $NCIRCLE=51; " Number of segments to use in drawing circles"
134 REAL*4 XCYL($NCIRCLE),YCYL($NCIRCLE),ZCYL($NCIRCLE); " Used in object "

135 COMMON/PASSIT/IRDISC($MXREG),NR,NZ; "NR=number of cylinders, NZ=    "
136                                     " number of slabs "
```

```
137  "  Particle tracking declarations  "

138  INTEGER*4 NPT($MXSTACK); " Number of points stored in coord arrays "
139  INTEGER*4 IQPREV($MXSTACK); "Charges of particles on stack"
140  REAL*4 ENPREV($MXSTACK); " Energies of particles on stack "
141  REAL*4 XPT($MXSTACK,2), " Coords of particles are stored here until "
142         YPT($MXSTACK,2), " until we get around to plotting them  "
143         ZPT($MXSTACK,2);

144  "  User-accessible scaling information  "

145  COMMON/SHWRSC/XSCA   YSCALE,ZSCALE,SCFACT;  "  Mult. scale factors"
146  COMMON/SHOWRP/ENRGCT(6),  " Variables the user can change to "
147                XTOPLN,      " configure the plot to his needs  "
148                YTOPLN,
149                COLORS(3),TOPCON,TOPTXT,TOPCAS;
150  CHARACTER*8 COLORS;
151  CHARACTER*50 TOPCON,TOPTXT,TOPCAS;

152  "  IBM 5080 display event loop control info  "

153  CHARACTER*32 EVTSTR;
154  INTEGER*4  IEVT(3);
155  REAL*4  XEVT(128),YEVT(128);
156  INTEGER*4 KEYBRD/1/,PICK/2/,BUTTON/3/,STROKE/4/;

157  "  The next numbers are the values reported for the PF keys if a      "
158  "  button board is attached to the 5080.  These are just the PF key  "
159  "  value+32.  If a button board is not present, the keys will be      "
160  "  reported at their face value, e.g., PF03 is reported as 3.         "

161  INTEGER*4 PF03/35/,  "  Terminates further plots  "
162            PF04/36/,  "  Erase current plot, start new plot  "
163            PF05/37/;  "  Save current plot, start new plot  "

164  "  Scaling information  "

165  REAL*4 VWPRT(2,2)/0.0,0.0,1.0,1.0/;  "  2D viewport  "
166  REAL*4 WDOW(2,2) /0.0,0.0,100.0,100.0/;  "  2D window  "

167  REAL*4 WLDVOL(3,2); " 3D world volume "
168  REAL*4 OBJVOL(3,2); " 3D object volume  "
169  REAL*4 VPRT3D(2,2)/0.154,0.199,0.846,0.891/; " 3D viewport  "
170  " Original values: 0.115,0.160,0.885,0.930  (however, the   "
171  "                  new values make for easier photography). "

172  REAL*4 XBOX(5),YBOX(5);  "  Used to make boxes on the screen  "

173  "  Graphic segment identifiers and view flags  "

174  COMMON/IVIEW/IVWOBJ,IVWELC,IVWPHT,IVWPOS,IDGOBJ,IDGELC,IDGPHT,IDGPOS;

175  "  Pick identifiers  "
```

```
176  INTEGER*4 PIKOBJ/1/;
177  INTEGER*4 PIKELC/2/;
178  INTEGER*4 PIKPBT/3/;
179  INTEGER*4 PIKPDS/4/;

180  "  The graphic segments  "

181  COMMON/GRAPHS/ISEG,SEGMNT($SEGSIZE),OBJECT($OBJSIZE),ELCTRN($TRKSIZE),
182                PROTN($TRKSIZE),PGSTRN($TRKSIZE),ISOTOP($TRKSIZE);
183  "  ISEG is a pointer that indicates which segment should be cleared  "
184  "  in routine UGXERR if necessary  "

185  "  Device characteristics  "

186  INTEGER*4 INTACT(10);

187  "  Line characteristics  "

188  CHARACTER*8 BRTNES(5)/'    VDIM',   "  Brightness options  "
189                       '     DIM',
190                       '  MEDIUM',
191                       '  BRIGHT',
192                       ' VBRIGHT'/;

193  CHARACTER*26 QUALTY/'   SOLID,          ,          '/;  " Line character "

194  "  Viewability box characteristics and texts  "

195  REAL*4 XVWTOP/91.0/;  "  Coordinates of top left corner of first box  "
196  REAL*4 YVWTOP/85.0/;  "  in WDOW coordinate system  "
197  REAL*4 BOXDEL/8.0/;   "  Box width  "
198  REAL*4 BOXSEP/4.0/;   "  Vertical separation of boxes  "

199  CHARACTER*24 VWCTRL(4)/'BRIGHT,MAGENTA,PICKID= 1', " Object   view box "
200                         'BRIGHT,GREEN  ,PICKID= 2', " Electron view box "
201                         'BRIGHT,YELLOW ,PICKID= 3', " Photon   view box "
202                         'BRIGHT,RED    ,PICKID= 4'/;" Positron view box "
203  CHARACTER*24 VWFILL(4)/'VDIM  ,BLACK  ,PICKID= 1', " Object   view box "
204                         'VDIM  ,BLACK  ,PICKID= 2', " Electron view box "
205                         'VDIM  ,BLACK  ,PICKID= 3', " Photon   view box "
206                         'VDIM  ,BLACK  ,PICKID= 4'/;" Positron view box "
207  CHARACTER*4  VWTEXT(4)/'3OBJ',  "  Object label  "
208                         'E2-3',  "  Electron label  "
209                         '3G  ',  "  Photon label  "
210                         'E2+3'/; "  Positron label  "
211  CHARACTER*4  VWCASE(4)/'U LL',  "  Case control for labels  "
212                         'LX X',
213                         'UG  ',
214                         'LX X'/;
215  CHARACTER*15 VWSIZE  /'CENTER,SIZE=2.5'/;  "  Size of labels  "
216  CHARACTER*40 VWLABL;  "  Label control variable  "
217  CHARACTER*22 ENLABL(3)/'MEDIUM,GREEN ,SIZE=1.5', "  Energy legend label"
218                         'MEDIUM,YELLOW,SIZE=1.5', "  attributes  "
219                         'MEDIUM,RED   ,SIZE=1.5'/;
```

```
220  CHARACTER*12 STRING;  "  Used to convert energy cuts to characters  "

221  LOGICAL*1 QINTER;  "  T=we are using an interactive plotting device  "

222  LOGICAL*1 QNOPLT/.FALSE./;  "  Flag:  T=do not make shower plots  "
223                              "  unless it is an initialization call  "
224  LOGICAL*1 QFIRST/.TRUE./;

225  LOGICAL*1 QOVRPL/.FALSE./;  "  If set true, allows overplotting of  "
226                              "  tracks from more than one shower  "

227  INTEGER*4 TXTEND; " Position of last non-blank character in title text "

228  COMMON/PLAIN/QPLAIN;          "  T=plain picture (default in BLOCK DATA "
229                                "  F=5080 borders/titles/icons (will be "
230                                "    set to .FALSE. (below) if IDEV = 2 "
231  LOGICAL*1 QPLAIN;

232  COMMON/SHOWGF/IPLOT;

233  COMIN/RANDOM/; "Located here to avoid FORTRAN 77 diagnostic message"

234  DATA IDEV/1/;          "1=PDEVLIN (default), 2=IBM5080"
235  "Note: IDEV currently gets selected below depending on the IPLOT-value"

236  "-------------------------------------------------------------------"
237  "  Executable code starts here                                      "
238  "-------------------------------------------------------------------"

239  IF(QNOPLT) RETURN;  "  Force initialization call first  "

240  "==================================================================="
241  GO TO (:Init_Shower:,:Add_Track:,:5080_Local_Control:,:Term_Job:,
242         :Plot_Isotope:) ISHOW;  "  Take action requested  "
243  "==================================================================="

244  "==========="
245  :Init_Shower:  "  Start a new shower plot  "
246  "==========="

247  "==================================================================="
248  IF(QOVRPL.OR.IARGD.EQ.-1) GO TO :Over_Plot:;   " Only do part of the   "
249                                                 " initialization if we are "
250                                                 " overplotting showers     "
251  "==================================================================="

252  IF(QFIRST) [  "  Once-per-job initializations  "

253    QFIRST=.FALSE.;

254    "-----------------------------------------------------------------"
255    "  Open the graphic device (IPLOT overrides DATA IDEV/1/ above).   "
256    "-----------------------------------------------------------------"
257    "   PDEVLIN         IBM5080  "
```

```
258     IF(IPLOT.EQ.1.OR.IPLOT.EQ.2) [IDEV=IPLOT;]
259     ELSE [
260       OUTPUT IPLOT;
261       (' *** STOPPED IN SHOWPL WITH IPLOT=',I3,' (I.E., NOT 1 OR 2)');
262       STOP;
263       ]

264     IF(IDEV.EQ.1) [
265       CALL UGOPEN('PDEVLIN,FLAG=99,XMAX=100000,YMAX=100000',1);
266     ]
267     ELSEIF(IDEV.EQ.2) [
268       CALL UGOPEN('IBM5080,DDNAME=IBM5080,NOPICKO,EXTKEYS,
269         VIEWCTL=3DDIAL',1);
270       QPLAIN=.FALSE.;
271       ]
272     ELSE [
273       OUTPUT IDEV;
274       (' *** STOPPED IN SHOWPL WITH IDEV=',I3,' (I.E., NOT 1 OR 2)');
275       STOP;
276       ]
277     "-----------------------------------------------------------------"

278     CALL UGINFO('ILEVEL',JUNK,INTACT);  "  Is it interactive?   "
279                                         "  1=non-interactive,   "
280                                         "  2=slave display,     "
281                                         "  3=fully interactive  "
282     QINTER=.NOT.(INTACT(1).EQ.1 .OR. INTACT(1).EQ.2);

283     "  Set the 2D screen drawing space limits  "

284     CALL UGDSPC('PUT',1.0,1.0,1.0); " Exactly square "

285     CALL UGFONT('DUPLEX');  "  Use duplex font  "

286     ]

287   CALL UGPICT('CLEAR',0);  "  Clear the entire display  "

288   ISEG=1;  "  Set UGXERR segment pointer to 2D segment first  "

289   "  Initialize all graphic segments  "

290   CALL UGINIT('CLEAR',SEGMNT,$SEGSIZE);  "  General graphic segment  "
291   CALL UGINIT('CLEAR',OBJECT,$OBJSIZE);  "  Object display segment   "
292   CALL UGINIT('CLEAR',ELCTRN,$TRKSIZE);  "  Electron tracks segment  "
293   CALL UGINIT('CLEAR',PHOTN ,$TRKSIZE);  "  Photon tracks segment    "
294   CALL UGINIT('CLEAR',POSTRN,$TRKSIZE);  "  Positron tracks segment  "
295   CALL UGINIT('CLEAR',ISOTOP,$TRKSIZE);  "  Isotope positions segment  "

296   "  Initialize segment identifiers  "

297   IDGOBJ=1000;  "  The object  "
298   IDGELC=2000;  "  Electron tracks  "
299   IDGPHT=3000;  "  Photon tracks  "
```

```
300  IDGPOS=4000;  "  Positron tracks  "

301  "  Turn on all particle viewing flags: 0=turn off tracks, 1=turn on "

302  IVWOBJ=1;  "  The object  "
303  IVWELC=1;  "  Electrons  "
304  IVWPHT=1;  "  Photons    "
305  IVWPOS=1;  "  Positrons  "

306  "  Set up 2D and 3D scaling based on object size  "

307  CALL UGWDOW('PUT',VWPRT,WDOW);  "  2D scale of screen  "

308  "-------------------------------------------------------------------------"
309  "  Draw the screen border                                                 "
310  "-------------------------------------------------------------------------"

311  XBOX(1)=WDOW(1,1); YBOX(1)=WDOW(2,1);
312  XBOX(2)=WDOW(1,1); YBOX(2)=WDOW(2,2);
313  XBOX(3)=WDOW(1,2); YBOX(3)=WDOW(2,2);
314  XBOX(4)=WDOW(1,2); YBOX(4)=WDOW(2,1);
315  XBOX(5)=WDOW(1,1); YBOX(5)=WDOW(2,1);

316  IF(.NOT.QPLAIN) CALL UGPLIN('SOLID,BRIGHT,BLUE',XBOX,YBOX,5,1,1,SEGMNT);

317  "-------------------------------------------------------------------------"
318  "  Draw the 3D viewport boundary                                          "
319  "-------------------------------------------------------------------------"

320  XBOX(1)=100.0*VPRT3D(1,1); YBOX(1)=100.0*VPRT3D(2,1);
321  XBOX(2)=100.0*VPRT3D(1,1); YBOX(2)=100.0*VPRT3D(2,2);
322  XBOX(3)=100.0*VPRT3D(1,2); YBOX(3)=100.0*VPRT3D(2,2);
323  XBOX(4)=100.0*VPRT3D(1,2); YBOX(4)=100.0*VPRT3D(2.1);
324  XBOX(5)=100.0*VPRT3D(1,1); YBOX(5)=100.0*VPRT3D(2,1);

325  IF(.NOT.QPLAIN) CALL UGPLIN('SOLID,BRIGHT,CYAN',XBOX,YBOX,5,1,1,SEGMNT);

326  "  Make the object occupy more of the plotting area  "

327  IF(QPLAIN) [
328    VPRT3D(1,1)=0.0; VPRT3D(2,1)=0.0;
329    VPRT3D(1,2)=1.0; VPRT3D(2,2)=1.0;
330    ]
331  CALL HOWPL(OBJVOL,WLDVOL,VPRT3D);  "Scale and draw the object volume"

332  ISEG=1;  "  Set UGXERR segment pointer back to 2D segment  "

333  "  Write out the user's top line text  "

334  ITXTL=LEN(TOPTXT);
335  DO ICHAR=ITXTL,1,-1 [
336    IF(TOPTXT(ICHAR:ICHAR)=' ') NEXT;
337    EXIT;
338    ]
```

```
339  TXTEND=ICHAR;  "  Position of last non-blank character in title text  "

340  IF(.NOT.QPLAIN) CALL UGXTXT(TOPCON,XTOPLN,YTOPLN,TOPTXT(1:TXTEND),
341    TOPCAS(1:TXTEND),SEGMNT);

342  CALL UGWRIT(' ',0,SEGMNT);  "  Write out anything up to this point  "
343                              "  since now we need to turn on pickability"
344  CALL UGINIT('CLEAR',SEGMNT,$SEGSIZE);  "  and clear the segment  "

345  "  Draw the viewability toggle switches on the screen for the object  "
346  "  and the tracks  "

347  DO IVW=1,4 [ "  Loop over the four viewability boxes  "
348    DO IMARGN=1,2 [  "  Put them in both left and right margins  "

349      "  Draw the box  "

350      IF(IMARGN = 1) [ XBOX(1)=XVWTOP; ]  "  Right-hand margin  "
351      ELSE [ XBOX(1)=100.0-XVWTOP-BOXDEL; ]  "  Left-hand margin  "

352      YBOX(1)=YVWTOP-FLOAT(IVW-1)*(BOXDEL+BOXSEP);

353      XBOX(2)=XBOX(1)+BOXDEL; YBOX(2)=YBOX(1);
354      XBOX(3)=XBOX(2);        YBOX(3)=YBOX(2)-BOXDEL;
355      XBOX(4)=XBOX(1);        YBOX(4)=YBOX(3);
356      XBOX(5)=XBOX(1);        YBOX(5)=YBOX(1);

357      IF(QDBGPL) [
358        OUTPUT IVW,VWCTRL(IVW); ('0','IVW=',I1,2X,'VWCTRL=[',A24,']');
359        ]

360      IF(.NOT.QPLAIN) [
361        CALL UGPFIL(VWFILL(IVW),XBOX,YBOX,5,SEGMNT);
362        CALL UGPLIN(VWCTRL(IVW),XBOX,YBOX,5,1.1,SEGMNT);
363        ]

364      "  Put the text in the box  "

365      "  Update PICK segment ID  "

366      VWLABL(1:24)=VWCTRL(IVW)(1:24);  "  Construct label control  "
367      VWLABL(25:25)=',';
368      VWLABL(26:40)=VWSIZE;

369      IF(QDBGPL) [
370        OUTPUT IVW,VWLABL; ('0','IVW=',I1,2X,'VWLABL=[',A40,']');
371        ]

372      IF(.NOT.QPLAIN) CALL UGXTXT(VWLABL,XBOX(1)+0.5*BOXDEL,
373        YBOX(1)-0.5*BOXDEL,VWTEXT(IVW),VWCASE(IVW),SEGMNT);

374    ]

375    "  Must use one UGINIT/UGWRIT pair for each PICKID to make boxes  "
```

```
376       "  in opposite margins highlight together  "

377       IF(IDEV.EQ.1) ["For PDEVLIN device"
378         CALL UGWRIT('     ',IVW,SEGMNT); " Make the view boxes pickable  "
379         ]
380       ELSEIF(IDEV.EQ.2) ["For IBM5080 device"
381         CALL UGWRIT('PICK',IVW,SEGMNT); " Make the view boxes pickable  "
382         ]
383       CALL UGINIT('CLEAR',SEGMNT,$SEGSIZE);  "  and clear the segment  "

384     ]

385   "  Write out PF key definitions  "

386   IF(.NOT.QPLAIN) [
387     CALL UGTEXT('SIZE=1.5,CYAN,BRIGHT',2.5,13.0,
388                 'PF03=QUIT FURTHER SHOWERS',SEGMNT);
389     CALL UGTEXT('SIZE=1.5,CYAN,BRIGHT',2.5,10.5,
390                 'PF04=ERASE THIS SHOWER, START ANOTHER',SEGMNT);
391     CALL UGTEXT('SIZE=1.5,CYAN,BRIGHT',2.5, 8.0,
392                 'PF05=KEEP THIS SHOWER, START ANOTHER',SEGMNT);
393     ]

394   "  Plot the energy-intensity legend  "

395   YCE=13.0;       "  Write out energy cut values  "

396   XEN=60.0; YEN=YCE-3.0; ELEN=5.0; EXSEP=1.0; EYSEP=2.0;

397   DO INTNS=2,5 [
398     XCE=XEN+FLOAT(INTNS-2)*(ELEN+EXSEP)+0.5*ELEN;
399     CALL UGCNVF(ENRGCT(INTNS),0,STRING,NBL);  "  Convert en cut to char  "
400     IF(.NOT.QPLAIN) CALL UGTEXT('BRIGHT,CYAN,SIZE=1.5,CENTER',
401                                 XCE,YCE,STRING(13-NBL:12),SEGMNT);
402     ]

403   CALL UGCNVF(ENRGCT(5),0,STRING,NBL);  "  Convert en cut to char  "
404   STRING(13-NBL-1:13-NBL-1)='>';
405   NBL=NBL+1;
406   XCE=XEN+4.0*(ELEN+EXSEP)+0.5*ELEN;
407   IF(.NOT.QPLAIN) CALL UGTEXT('BRIGHT,CYAN,SIZE=1.5,CENTER',
408                               XCE,YCE,STRING(13-NBL:12),SEGMNT);

409   IF(.NOT.QPLAIN) CALL UGXTXT('MEDIUM,CYAN,SIZE=1.5,CENTER',
410                               XCE+4.5,YCE,'MEV',' L ',SEGMNT);


411   DO ICOLOR=1,3 [  "  Draw the line segments  "
412     DO INTNS=1,5 [
413       XCE=XEN+FLOAT(INTNS-1)*(ELEN+EXSEP);
414       YCE=YEN-FLOAT(ICOLOR-1)*EYSEP;
415       QUALTY(10:17)=BRTNES(INTNS)(1:8);
416       QUALTY(19:26)=COLORS(ICOLOR)(1:8);
417       IF(.NOT.QPLAIN) CALL UGLINE(QUALTY,XCE,YCE,0,SEGMNT);
```

```
418       XCE=XCE+ELEN;
419       IF(.NOT.QPLAIN) CALL UGLINE(QUALTY,XCE,YCE,1,SEGMNT);
420       ]
421     ]

422   IF(.NOT.QPLAIN) [
423     CALL UGXTXT(ENLABL(1),XCE+1.5*ELEN,YEN,'E2-3','LX X',SEGMNT);
424     CALL UGXTXT(ENLABL(2),XCE+1.5*ELEN,YEN-EYSEP,'G','G',SEGMNT);
425     CALL UGXTXT(ENLABL(3),XCE+1.5*ELEN,YEN-2.0*EYSEP,'E2+3',
426                 'LX X',SEGMNT)
427     ]

428   CALL UGWRIT(' ',0,SEGMNT);  "  Write out 2D info  "
429   CALL UGINIT('CLEAR',SEGMNT,$SEGSIZE);  "  and clear the segment  "

430   "=========="
431   :Over_Plot:
432   "=========="

433   QOVRPL=.FALSE.;  "  Reset flag  "

434   "  Initialize pointers for tracking  "

435   DO IPT=1,$MXSTACK [ NPT(IPT)=0; "  No points stored yet  " ]

436   RETURN;

437   "=========="
438   :Add_Track:  "  Add a vector to a track  "
439   "=========="

440   " Commenting out.....
441   OUTPUT IARGD,NP,IQ(NP),E(NP),Z(NP),W(NP);
442   (' ','Entering:  IARG,NP,IQ,E,Z,W=',3I3,2X,3F12.4);

443   OUTPUT NPT(NP),ZPT(NP,NPT(NP)),IQPREV(NP),ENPREV(NP);
444   (' ','NPT(NP), ZPT(NP,NPT(NP)), IQPREV(NP), ENPREV(NP)=',
445   I2,1X,F10.5,1X,I2
446   ,F10.4);
447   ..  ."

448   XSCLD=X(NP)*XSCALE;  "  Scale each coordinate  "
449   YSCLD=Y(NP)*YSCALE;
450   ZSCLD=Z(NP)*ZSCALE;

451   NPT(NP)=NPT(NP)+1;
452   IF(NPT(NP).EQ.1) [ IQPREV(NP)=IQ(NP); ENPREV(NP)=E(NP); ]
453   XPT(NP,NPT(NP))=XSCLD;
454   YPT(NP,NPT(NP))=YSCLD;
455   ZPT(NP,NPT(NP))=ZSCLD;
456   "
457   OUTPUT NPT(NP),ZPT(NP,NPT(NP)),IQPREV(NP),ENPREV(NP);
458   (' ','NPT(NP), ZPT(NP,NPT(NP)), IQPREV(NP), ENPREV(NP)='
459   ,I2,1X,F10.5,1X,I2
```

```
460  ,F10.4);
461  "
462  IF(NPT(NP) = 2) [  "  We need to plot a vector  "

463  "OUTPUT; (' ','Plotting a segment');"
464    "  Determine line quality parameters  "

465    ENPRKE=ENPREV(NP)-IABS(IQPREV(NP))*PRM;  "Kinetic energy (MeV)"
466    IF     (ENPRKE.GE.ENRGCT(6)) [ INTNST=0; ]  "  No plot     "
467    ELSEIF(ENPRKE.GE.ENRGCT(5)) [ INTNST=6; ]  "  Brightest  "
468    ELSEIF(ENPRKE.GE.ENRGCT(4)) [ INTNST=4; ]
469    ELSEIF(ENPRKE.GE.ENRGCT(3)) [ INTNST=3; ]
470    ELSEIF(ENPRKE.GE.ENRGCT(2)) [ INTNST=2; ]
471    ELSEIF(ENPRKE.GE.ENRGCT(1)) [ INTNST=1; ]  "  Dimmest  "
472    ELSE                          [ INTNST=0; ]  "  No plot     "

473    IF(INTNST = 0) GO TO :Plot_Done:;  "  Skip the plotting ?  "

474    QUALTY(10:17)=BRTNES(INTNST)(1:8);
475    QUALTY(19:26)=COLORS(IQPREV(NP)+2)(1:8);

476    "  Plot the vector  "

477    IF(IQPREV(NP).EQ.-1) [
478      CALL CHKBUF(ELCTRN,IDGELC,$TRKSIZE);  "  Make room for this track "
479      "  First move to start of line segment  "
480      CALL UG3LIN(QUALTY,XPT(NP,1),YPT(NP,1),ZPT(NP,1),0,ELCTRN);
481      "  Then plot to the end of the line segment "
482      CALL UG3LIN(QUALTY,XPT(NP,2),YPT(NP,2),ZPT(NP,2),1,ELCTRN);
483      ]
484    ELSEIF(IQPREV(NP).EQ.0) [
485      CALL CHKBUF(PHOTN ,IDGPHT,$TRKSIZE);
486  "  2DMOD RFC 870602  "
487      CALL UG3LIN(QUALTY,XPT(NP,1),YPT(NP,1),ZPT(NP,1),0,PHOTN );
488      CALL UG3LIN(QUALTY,XPT(NP,2),YPT(NP,2),ZPT(NP,2),1,PHOTN );
489  "  END-2DMOD  "
490      ]
491    ELSE [
492      CALL CHKBUF(POSTRN,IDGPOS,$TRKSIZE);
493      CALL UG3LIN(QUALTY,XPT(NP,1),YPT(NP,1),ZPT(NP,1),0,POSTRN);
494      CALL UG3LIN(QUALTY,XPT(NP,2),YPT(NP,2),ZPT(NP,2),1,POSTRN);
495      ]

496   "==========="
497    :Plot_Done:
498   "==========="

499    ]  "End of plotting loop"

500    "  Update the arrays and pointers for the next track segment  "

501  IF(IARGC > 0) [
502  NPT(NP)=0;
503  "
```

```
504  OUTPUT NP,NPT(NP);(' ','Discard: NPT(',I2,')=',I2);
505  "
506  ] "Particle is discarded"
507  ELSEIF(NPT(NP) = 2) [  "  Particle was plotted and we must wrap  "
508    XPT(NP,1)=XPT(NP,2); "  around to the beginnning  "
509    YPT(NP,1)=YPT(NP,2);
510    ZPT(NP,1)=ZPT(NP,2);
511    IQPREV(NP)=IQ(NP);
512    ENPREV(NP)=E(NP);
513    NPT(NP)=1;
514  "
515  OUTPUT NPT(NP),ZPT(NP,NPT(NP)),IQPREV(NP),ENPREV(NP);
516  (' ','Wrapping:',/,' ',
517  'NPT(NP), ZPT(NP,NPT(NP)), IQPREV(NP), ENPREV(NP)=',
518  I2,1X,F10.5,1X,I2
519  ,F10.4);"
520    ]

521  "
522  OUTPUT;(' ','------------------Leaving Add_Track---------------');
523  "
524  RETURN;


525  "============="
526  :Plot_Isotope:  "  Plot an isotope position  "
527  "============="

528  "  Draw a point at the current position  "

529  $RANDOMSET FINPOS;
530  FINPOS=FINPOS*TVSTEP;
531  $FINVAL(FINPOS,XF,YF,ZF);
532  XF=XF*XSCALE;  "  Scale each coordinate  "
533  YF=YF*YSCALE;
534  ZF=ZF*ZSCALE;

535  CALL UG3MRK('VBRIGHT,WHITE',XF,YF,ZF,ISOTOP);

536  RETURN;

537  "==================="
538  :5080_Local_Control:  "  Process user requests from the 5080 display  "
539  "==================="
540                        "  until he is through with this shower  "

541  IDGELC=IDGELC+1;  "  Get new identifier for each graphic segment  "
542  IDGPHT=IDGPHT+1;
543  IDGPOS=IDGPOS+1;

544  CALL UGWRIT(' ',IDGELC,ELCTRN);  "  Write out remaining tracks  "
545  CALL UGWRIT(' ',IDGPHT,PHOTN );
546  CALL UGWRIT(' ',IDGPDS,POSTRN);
547  CALL UGWRIT(' ',0     ,ISDTOP);
```

```
548  CALL UGINIT('CLEAR',ELCTRN,$TRKSIZE);  "  and clear the segments  "
549  CALL UGINIT('CLEAR',PHOTN ,$TRKSIZE);
550  CALL UGINIT('CLEAR',POSTRN,$TRKSIZE);
551  CALL UGINIT('CLEAR',ISOTOP,$TRKSIZE);

552  "  Enter the 5080 event loop  "

553  "  5080 display PF key usage:                                          "
554  "  --------------------------                                          "
555  "                                                                      "
556  "     PF03 - Stop generating shower plots                              "
557  "     PF04 - Erase current event, generate new one                     "
558  "     PF05 - Save current event, generate new one                      "

559  "==============="
560  :Wait_for_Event:
561  "==============="

562  IF(.NOT.QINTER) GO TO :Exit_Event_Loop::

563  CALL UGEVNT(' ',-1.0,EVTSTR,IEVT,XEVT,YEVT);  "  Wait for user event  "
564  "OUTPUT IEVT(1),IEVT(2),IEVT(3);
565  ('0','IEVT = ',3I5); "

566  "  Check to see if a PF key was selected, in either standard or  "
567  "  extended reporting format.  "

568  IF(IEVT(1) = BUTTON & ( ( IEVT(2) = PF03 ) | ( IEVT(2) = 3 ) ) )  [
569    GO TO :No_More_Plots:;
570    ]
571  ELSEIF(IEVT(1) = BUTTON & ( ( IEVT(2) = PF04 ) | ( IEVT(2) = 4 ) ) )  [
572    GO TO :Exit_Event_Loop:;
573    ]
574  ELSEIF(IEVT(1) = BUTTON & ( ( IEVT(2) = PF05 ) | ( IEVT(2) = 5 ) ) )  [
575    QOVRPL=.TRUE.;
576    GO TO :Exit_Event_Loop:;
577    ]
578  ELSEIF(IEVT(1) = PICK) [  "  A PICK event occurred  "
579    IPICK=IEVT(3);  "  Get final PICK id for object, electrons,  "
580                    "  photons, or positrons  "
581    IF(QDBGPL) [
582      OUTPUT IEVT(3),IPICK; ('0','IEVT(3)=',I3,2X,'IPICK=',I3);
583      ]
584    IF(IPICK = PIKOBJ) [  "  Toggle object viewability  "
585      IF(QDBGPL) [ OUTPUT IVWOBJ,IDGOBJ; (' ','OBJ:',2X,2I5); ]
586      IF(IVWOBJ = 1) [ IVWOBJ=0; CALL OMTSEG('OMIT',IDGOBJ); ]
587      ELSE           [ IVWOBJ=1; CALL OMTSEG('INCLUDE',IDGOBJ); ]
588      ]
589    ELSEIF(IPICK = PIKELC) [  "  Toggle electron viewability  "
590      IF(QDBGPL) [ OUTPUT IVWELC,IDGELC; (' ','ELC:',2X,2I5); ]
591      IF(IVWELC = 1) [ IVWELC=0; CALL OMTSEG('OMIT',IDGELC); ]
592      ELSE           [ IVWELC=1; CALL OMTSEG('INCLUDE',IDGELC); ]
593      ]
```

```
    ELSEIF(IPICK = PIKPHT) [  "  Toggle photon viewability  "
      IF(QDBGPL) [ OUTPUT IVWPHT,IDGPHT; (' ','PHT:',2X,2I5); ]
      IF(IVWPHT = 1) [ IVWPHT=0; CALL OMTSEG('OMIT',IDGPHT); ]
      ELSE           [ IVWPHT=1; CALL OMTSEG('INCLUDE',IDGPHT); ]
      ]
    ELSEIF(IPICK = PIKPOS) [  "  Toggle positron viewability  "
      IF(QDBGPL) [ OUTPUT IVWPOS,IDGPOS; (' ','POS:',2X,2I5); ]
      IF(IVWPOS = 1) [ IVWPOS=0; CALL OMTSEG('OMIT',IDGPOS); ]
      ELSE           [ IVWPOS=1; CALL OMTSEG('INCLUDE',IDGPOS); ]
      ]
    ELSE [
      IF(QDBGPL) [ OUTPUT; (' ','Unknown PICK event'); ]
      ]
    GO TO :Wait_for_Event:;  "  Go get another event  "
    ]
  ELSE [ GO TO :Wait_for_Event:; ]  "  Unknown event--ignore it  "

  "================"
  :Exit_Event_Loop:
  "================"

  RETURN;

  "==============="
  :No_More_Plots:  "  Do not make any more shower plots  "
  "==============="

  QNOPLT=.TRUE.;

  "========"
  :Term_Job:  "  Terminate the plotting--no more showers to plot  "
  "========"

  OUTPUT; ('0','Closing all plotting devices');
  CALL UGCLOS('ALL,NOCLEAR');

  RETURN;

  END;

  %E
  "*************************************************************************"
  "                                    STANFORD LINEAR ACCELERATOR CENTER"
  BLOCK DATA;
  "                                    EGS4 SUBPROGRAM -  5 JUN 1987/1500"
  "*************************************************************************"
  "   For: SHOWPL-HOWPL                                                     "

  COMMON/SHWRSC/XSCALE,YSCALE,ZSCALE,SCFACT;  "  Mult. scale factors"
  COMMON/SHOWRP/ENRGCT(6),  " Variables the user can change to "
                XTOPLN,     " configure the plot to his needs  "
                YTOPLN,
                COLORS(3),TOPCOM,TOPTXT,TOPCAS;
```

```
637  COMMON/PLAIN/QPLAIN;          "  T=plain picture (default set below) "
638                                "  F=5080 borders/titles/icons "
639  LOGICAL*1 QPLAIN;

640  CHARACTER*8 COLORS;
641  CHARACTER*50 TOPCON,TOPTXT,TOPCAS;

642  DATA QPLAIN/.TRUE./;  " Default is for plain picture "

643  DATA XSCALE/1.0/,  "  Default is unity scaling  "
644       YSCALE/1.0/,
645       ZSCALE/1.0/;

646  DATA SCFACT/2.5/;  "Scale factor between world and object volumes"
647                     " (affects zooming capability)"

648  DATA ENRGCT/0.0,1.0,5.0,10.0,50.0,1.0E8/; " Brightness control  "
649  "  If K.E.  < ENRGCT(1), particle is not plotted; if K.E. > ENRGCT(1), "
650  "  plot it as VDIM, etc.  If K.E.  > ENRGCT(6), do not plot it.  "

651  DATA COLORS/'   GREEN',   "  Electron color  "
652              ' YELLOW',   "  Photon color    "
653              '     RED'/;  "  Positron color  "

654  "  Top line text  "
655  DATA TOPTXT  /'EGS4 CYLINDER/SLAB GEOMETRY                          '/;
656  "  Top line case control  "
657  DATA TOPCAS  /'      LLLLLL  LLL  LLLLLL                            '/;
658  "  Coordinates of top line text  "
659  DATA XTOPLN/50.0/,YTOPLN/97.0/;
660  "  Top line characteristics  "
661  DATA TOPCON  /'CENTER,SIZE=3.0,CYAN                                 '/;

662  END;

663  %E
664  "*********************************************************************"
665  "                                  STANFORD LINEAR ACCELERATOR CENTER"
666  SUBROUTINE UGXERR(LEVEL,NAME,INDEX);
667  "                                  EGS4 SUBPROGRAM - 31 AUG 1988/0000"
668  "*********************************************************************"
669  COMMON/GRAPHS/ISEG,SEGMNT($SEGSIZE),OBJECT($OBJSIZE),ELCTRN($TRKSIZE),
670                PHOTN($TRKSIZE),POSTRN($TRKSIZE),ISOTOP($TRKSIZE);
671  CHARACTER*8 NAME;

672  "IF THE ERROR IS AN INDICATION THAT NO MORE SPACE IS AVAILABLE IN"
673  "THE GRAPHIC ELEMENT, THEN THE CURRENT CONTENTS OF THE ELEMENT ARE"
674  "TRANSMITTED TO THE GRAPHIC DEVICE, THE ELEMENT IS RE-INITIALIZED,"
675  "AND THE ERROR INDICATOR IS RE-SET."

676  IF(INDEX.EQ.11) [  "  A graphic segment is full--clear it and reset  "
677    IF(ISEG.EQ.1) [  "  Clear the 2D segment  "
678      CALL UGWRIT(' ',0,SEGMNT);
679      CALL UGINIT('CONTINUE',SEGMNT,$SEGSIZE);
```

```
680       LEVEL=0;  "  Reset error level to show it's been fixed  "
681       ]
682     ELSEIF(ISEG.EQ.2) [  "  Clear the 3D object segment  "
683       CALL UGWRIT(' ',1,OBJECT);  "  Pick ID of object is 1  "
684       CALL UGINIT('CONTINUE',OBJECT,$OBJSIZE);
685       LEVEL=0;  "  Reset error level to show it's been fixed  "
686       ]
687     ELSEIF(ISEG.EQ.3) [  "  Clear the isotope segment  "
688       CALL UGWRIT(' ',0,ISOTOP);  "  Pick ID of object is 1  "
689       CALL UGINIT('CONTINUE',ISOTOP,$TRKSIZE);
690       LEVEL=0;  "  Reset error level to show it's been fixed  "
691       ]
692     ELSE [  "  Unknown segment -- assume it's HANDYPAK  "
693       CALL DGPUTX(0);  "  HANDYPAK routine to flush its graphics buffer  "
694       LEVEL=0;
695       ]
696     ]
697   "  2DMOD RFC 870602  "
698   "  Removed code that was here  "
699   "  END-2DMOD  "
700   ELSE [
701     OUTPUT LEVEL,NAME,INDEX;(10X,'*** ERROR FROM UGXERR ***',I5,2X,A8,I5);
702     ]

703   RETURN;
704   END; "END OF SUBROUTINE UGXERR"

705   %E
706   "*************************************************************************"
707   "                                    STANFORD LINEAR ACCELERATOR CENTER"
708   SUBROUTINE CHKBUF(SEGM,SEGID,SEGSIZ);
709   "                                    EGS4 SUBPROGRAM -  5 JUN 1987/1500"
710   "*************************************************************************"
711   INTEGER SEGM(*);
712   INTEGER SEGID,SEGSIZ;
713   "  Clear out a graphic segment if necessary  "
714   IF(SEGM(1).LE.0.85*SEGSIZ) RETURN;
715   SEGID=SEGID+1;  "  Increment segment identifier for OMIT/INCLUDE use  "
716   CALL UGWRIT(' ',SEGID,SEGM);
717   CALL UGINIT('CONTINUE',SEGM,SEGSIZ);
718   RETURN;
719   END;

720   %E
721   "*************************************************************************"
722   "                                    STANFORD LINEAR ACCELERATOR CENTER"
723   SUBROUTINE OMTSEG(CMD,IDMAX);
724   "                                    EGS4 SUBPROGRAM -  5 JUN 1987/1500"
725   "*************************************************************************"

726   "  Required by routine SHOWPL.  Omits/includes segments for object and "
727   "  tracks.  "

728   "  861116 RFC  Creation date.                                           "
```

```
729  CHARACTER*(*) CMD;

730  "  Omit or include all object, electron, photon, or positron graphic  "
731  "  segments based on values in variable CMD  "

732  IDMIN=1000*(IDMAX/1000)+1;
733  DO ID=IDMIN,IDMAX [ CALL UGPICT(CMD,ID); ]

734  RETURN;
735  END;

736  %E
737  "*************************************************************************"
738  "                                    STANFORD LINEAR ACCELERATOR CENTER"
739  SUBROUTINE UGXLIN(FLAG,DATA);
740  "                                    EGS4 SUBPROGRAM -  5 JUN 1987/1500"
741  "*************************************************************************"
742  "  Write the graphics data out in a device-independent form.              "
743  "  Note: UGXLIN is called by UGS77 to open the PDEVLIN pseudo-device.     "
744  "        UGXUGS is supplied in the TEXT file PDEVUGSX (see the LOAD       "
745  "        statement in EGS4IN EXEC).                                       "

746  INTEGER FLAG,DATA(*);

747  CALL UGXUGS(FLAG,DATA);
748  RETURN;
749  END;

750  %E
751  "*************************************************************************"
752  "                                    STANFORD LINEAR ACCELERATOR CENTER"
753  SUBROUTINE HOWPL(OBJVOL,WLDVOL,VPRT3D);
754  "                                    EGS4 SUBPROGRAM -  5 JUN 1987/1500"
755  "*************************************************************************"
756  "+-----------------------------------------------------------------------+"
757  "|                                                                       |"
758  "|  HOWPL MORTRAN --  Required by routine SHOWPL in order to:            |"
759  "|                      A) perform object scaling                        |"
760  "|                      B) perform object drawing.                       |"
761  "|                                                                       |"
762  "|  NOTE:  This version of HOWPL incorporates the geometry of the        |"
763  "|  -----  EGS4 user code UCCYSL MORTRAN, a cylinder/slab                |"
764  "|         arrangement.  It may be used with any user code utilizing     |"
765  "|         this geometry.  HOWPL may be modified for other geometries    |"
766  "|         by changing the object volume calculation and the object      |"
767  "|         drawing sections (look for them below).  HOWPL must follow    |"
768  "|         SHOWPL because of Mortran3 macro definitions in the latter.   |"
769  "|                                                                       |"
770  "|                                                                       |"
771  "|  Written by     R. F. Cowan                                           |"
772  "|  ----------     Laboratory for Nuclear Science                        |"
773  "|                 Massachusetts Institute of Technology                 |"
```

```
774  "|                  Cambridge, MA 02139                                  |"
775  "|                                                                       |"
776  "|          and     W. R. Nelson                                         |"
777  "|                  Stanford Linear Accelerator Center                   |"
778  "|                  Stanford, CA 94305                                   |"
779  "|                                                                       |"
780  "+-----------------------------------------------------------------------+"
781  ;COMIN/CYLDTA,PLADTA/; "  CONTAINS GEOMETRY INFO  "

782  COMMON/PLAIN/QPLAIN;          "  T=plain picture (default in BLOCK DATA "
783                                "  F=5080 borders/titles/icons "
784  LOGICAL*1 QPLAIN;

785  COMMON/SHWRSC/XSCALE,YSCALE,ZSCALE,SCFACT;  "  Mult. scale factors"
786  REAL*4 CYLRAD($MXCYLS);  "  Cylinder radii  "

787  PARAMETER $NCIRCLE=51; " Number of segments to use in drawing circles"
788  REAL*4 XCYL($NCIRCLE),YCYL($NCIRCLE),ZCYL($NCIRCLE); " Used in object "

789  COMMON/PASSIT/IRDISC($MXREG),NR,NZ; "NR=number of cylinders, NZ=    "
790                                      " number of slabs "
791  LOGICAL*1 QINTER;  "  T=we are using an interactive plotting device  "
792  INTEGER*4 INTACT(10);

793  REAL*4 WLDVOL(3,2); " 3D world volume "
794  REAL*4 OBJVOL(3,2); " 3D object volume  "
795  REAL*4 VPRT3D(2,2); " 3D viewport  "
796  REAL*4 EYEPT(3); "Eye point for the parallel/perspective transformation"
797  REAL*4 UPDIR(3)/0.0,1.0,0.0/;  "  Up direction on screen  "
798  REAL*4 PFLAG/0.00/;  "  Projection flag:  0 --> parallel projection,  "
799                       "                    >0 --> perspective projection "

800  "  3D axes coordinates and blanking bits  "

801  REAL*4      XAXIS(5)/5*0.0/,
802              YAXIS(5)/5*0.0/,
803              ZAXIS(5)/5*0.0/;
804  INTEGER*4   AXESBB/ZB0000000/;  "  Blanking bits  "

805  "  Graphic segment identifiers and view flags  "

806  COMMON/IVIEW/IVWOBJ,IVWELC,IVWPHT,IVWPOS,IDGOBJ,IDGELC,IDGPHT,IDGPOS;

807  "  The graphic segments  "

808  COMMON/GRAPHS/ISEG,SEGMNT($SEGSIZE),OBJECT($OBJSIZE),ELCTRN($TRKSIZE),
809               PHOTN($TRKSIZE),POSTRN($TRKSIZE),ISOTOP($TRKSIZE);

810  "  Calculate 3D world and object volumes and 3D viewport  "

811  "  First the object volume (in whatever units EGS4 is using)  "

812  "  ***** This calculation of the object volume (OBJVOL) depends *****  "
813  "  ***** on the use of the EGS4 cylinder/slab geometry in the   *****  "
```

```
814  "  ***** UCCYSL user code or a similar user code and must be    *****  "
815  "  ***** modified if a different geometry is used.              *****  "

816  NRADII=NR-1;  "  Number of radii  "
817  DO IRAD=1,NRADII [
818    CYLRAD(IRAD)=SQRT(CYRAD2(IRAD));  "  Cylinder radii  "
819    ]

820  RADMAX=CYLRAD(NRADII);  "  Cylinder maximum radius  "
821  XLO=-RADMAX*XSCALE;
822  YLO=-RADMAX*YSCALE;
823  ZLO=PCOORD(3,1)*ZSCALE;
824  XHI=RADMAX*XSCALE;
825  YHI=RADMAX*YSCALE;
826  ZHI=PCOORD(3,NZ)*ZSCALE;
827  OBJVOL(1,1)=XLO;
828  OBJVOL(2,1)=YLO;
829  OBJVOL(3,1)=ZLO;
830  OBJVOL(1,2)=XHI;
831  OBJVOL(2,2)=YHI;
832  OBJVOL(3,2)=ZHI;

833  "  ***** End of object volume (OBJVOL) calculation.             *****  "

834  "-------------------------------------------------------------------------"
835  "-------- User should not change the following section of code --------"
836  "-------------------------------------------------------------------------"

837  "  Make the world volume bigger than the object volume  "

838  VAL=0.0;
839  DO IC=1,3 [
840    VAL=AMAX1(VAL,OBJVOL(IC,2)-OBJVOL(IC,1));
841    ]

842  DO IC=1,3 [
843    CTR=0.5*(OBJVOL(IC,1) + OBJVOL(IC,2));
844    WLDVOL(IC,1)=CTR - 0.5*VAL*SCFACT;
845    WLDVOL(IC,2)=CTR + 0.5*VAL*SCFACT;
846  ]

847  "  Calculate the eyepoint  "
848  "The first is for end-on and the second is for sideways view"
849  "EYEPT(1)=0.0; EYEPT(2)=0.0; EYEPT(3)=WLDVOL(3,2);"
850  EYEPT(1)=WLDVOL(1,1); EYEPT(2)=0.0;
851  EYEPT(3)=(WLDVOL(3,1) + WLDVOL(3,2))/2;

852  CALL UG3WRD('PUT',VPRT3D,WLDVOL);
853  CALL UG3TRN('PUT',OBJVOL,EYEPT,UPDIR,PFLAG);
854  CALL UGINFO('ILEVEL',JUNK,INTACT);  "  Is it interactive?    "
855                                      "  1=non-interactive,    "
856                                      "  2=slave display,      "
857                                      "  3=fully interactive   "
858  QINTER=.NOT.(INTACT(1).EQ.1 .OR. INTACT(1).EQ.2);
```

```
859  IF(QINTER) [
860    CALL UGENAB('KEYBOARD,PICK,BUTTON,STROKE');" Enable local input from "
861    ]                                          " the 5080 display unit   "
862  "-------------------------------------------------------------------------"

863  "  ***** This construction of the object outlines depends       *****  "
864  "  ***** on the use of the EGS4 cylinder/slab geometry in the   *****  "
865  "  ***** UCCYSL user code or a similar user code and must be    *****  "
866  "  ***** modified if a different geometry is used.              *****  "

867  "  Draw the outline of the cylinder/slab geometry  "

868  ISEG=2;  "  Set UGXERR segment pointer to the object segment while  "
869           "  we are drawing it so UGXERR can clear it every time we  "
870           "  manage to fill  it up  "

871  TWOPI=8.0*ATAN(1.0);
872  SEGNUM=FLOAT($NCIRCLE-1);
873  DO ICYL=1,NRADII [  "  Draw each cylindrical ring  "
874    DO IPLN=1,NZ  [  "  In each plane  "
875      DO IANG=1,$NCIRCLE [  "  if > 200 points makes a decent circle  "
876        ANGLE=TWOPI*FLOAT(IANG-1)/SEGNUM;
877        XCYL(IANG)=CYLRAD(ICYL)*COS(ANGLE)*XSCALE;
878        YCYL(IANG)=CYLRAD(ICYL)*SIN(ANGLE)*YSCALE;
879        ZCYL(IANG)=PCOORD(3,IPLN)*ZSCALE;
880        ]
881    CALL CHKBUF(OBJECT,IDGOBJ,$OBJSIZE);  "  Make room in the segment  "
882    CALL UG3PLN('MAGENTA,BRIGHT',XCYL,YCYL,ZCYL,$NCIRCLE,1,1,
883                OBJECT);
884      ]

885    DO IPLN=2,NZ [
886      CALL CHKBUF(OBJECT,IDGOBJ,$OBJSIZE);  "  Make room in the segment  "
887      IPLN1=0.0; XPLN2=XPLN1;
888      YPLN1=CYLRAD(ICYL)*YSCALE; YPLN2=YPLN1;
889      ZPLN1=PCOORD(3,IPLN-1)*ZSCALE; ZPLN2=PCOORD(3,IPLN)*ZSCALE;
890      CALL UG3LIN('MAGENTA,BRIGHT',XPLN1,YPLN1,ZPLN1,0,OBJECT);
891      CALL UG3LIN('MAGENTA,BRIGHT',XPLN2,YPLN2,ZPLN2,1,OBJECT);
892      YPLN1=-YPLN1; YPLN2=-YPLN2;
893      CALL UG3LIN('MAGENTA,BRIGHT',XPLN1,YPLN1,ZPLN1,0,OBJECT);
894      CALL UG3LIN('MAGENTA,BRIGHT',XPLN2,YPLN2,ZPLN2,1,OBJECT);

895      ]

896    ]

897  IF(.NOT.QPLAIN) [ "3D axes section"
898  "  Axis length is 1/10 of maximum object dimension "
899    AXISLN=AMAX1(   1.0,0.1*(OBJVOL(1,2)-OBJVOL(1,1)));
900    AXISLN=AMAX1(AXISLN,0.1*(OBJVOL(2,2)-OBJVOL(2,1)));
901    AXISLN=AMAX1(AXISLN,0.1*(OBJVOL(3,2)-OBJVOL(3,1)));

902    XAXIS(2)=AXISLN;
903    YAXIS(3)=AXISLN;
```

```
904     ZAXIS(5)=-AXISLN;

905   " Draw the axes "
906     CALL UG3PLN('CYAN ,BRIGHT',XAXIS,YAXIS,ZAXIS,5,AXESBB,-5,OBJECT);

907     AXL=1.15*AXISLN;  "  Text position slightly beyond end of axis  "

908     " Draw axes labels "
909     CALL UG3TXT('WHITE,BRIGHT',AXL,0.0,0.0,'X',OBJECT);
910     CALL UG3TXT('WHITE,BRIGHT',0.0,AXL,0.0,'Y',OBJECT);
911     CALL UG3TXT('WHITE,BRIGHT',0.0,0.0,-AXL,'-Z',OBJECT);
912     ]

913   IDGOBJ=IDGOBJ+1;  "  Increment object segment id  "
914   CALL UGWRIT(' ',IDGOBJ,OBJECT);  " Output remaining parts of object "
915   CALL UGINIT('CLEAR',OBJECT,$OBJSIZE);  "  and clear the object segment "

916   "  ***** End of object drawing code.                              *****  "

917   RETURN;
918   END;
```

Appendix B

EGS4PL Post-processor Code Listings

B.1. Building the EGS4PL processor

On VM/SP, the EGS4PL EXEC file compiles, links, parses options, and runs the processor each time. It requires that the Unified Graphics System libraries, input graphics file, the source code EGS4PL MORTRAN, and the EGS4 Mortran processor be available, in addition to itself.

On VAX/VMS, the DCL command file EGS4PL_B.COM (named for 'EGS4PL_Build', truncated to 8 characters) is executed once to install EGS4PL as a foreign command which is invoked at the DCL prompt. Note that this build file must be modified so it uses the correct directories on the local machine for the Unified Graphics System, EGS4 Mortran processor, *etc.*

B.2. Code Listings

Code Listing of the EGS4PL.EXEC Command File for VM/SP

```
1  /*
2    EGS4PL EXEC -- Runs EGS4PL MORTRAN to look at the output file
3                   from a 2D SHOWPL/HOWPL run produced via EGS4.
4    R. F. Cowan and W. R. Nelson 890922
5    Syntax:
6            EGS4PL fn ft fm ( options
7    where
8            fn = filename of 2D graphics file (no default; must be supplied)
9            ft = filetype of 2D graphics file (default PDEVLIN)
10           fm = filemode of 2D graphics file (default is first one found
11                                              in disk search order)
12   Options:  Allowed in any combination, any order.
13       SEQ4010   Selects graphics device.  Default is IMGN300.
14       IMPrt10
15       IMGn300
16       VEP12ff
17       IQoff chg chg chg     Turns off particles of type chg.
18       XZoom value  (default is 1.0)
19       YZoom value  (default is 1.0)
20        Zoom value  (default is 1.0)
```

```
21        XTrans value (default is 0.0, normally between -1.0 to +1.0)
22        YTrans value (default is 0.0, normally between -1.0 to +1.0)

23        BOX     Draw a border around the plotting area.
24  */

25  Trace Norm
26  Address Command

27  /*  Disk where all output goes--same as in EGS4IN EXEC  */

28  fmtdisk="A"

29  /*  Define some constants  */

30  true=1; false=0

31  /*  Initialize the zoom factors  */

32  xzvalue=1.0; yzvalue=1.0; zvalue=1.0

33  /*  Initialize translations about center of plot  */

34  xtrans=0.0; ytrans=0.0

35  /*  Initialize particle viewing flags: T = show it, F = omit it  */

36  electron="T"; photon="T"; positron="T"

37  /*  Initialize border plotting flag to false  */

38  box="F"

39  /*  Define the UGS77 names for the graphic devices  */

40  graphics_device="SEQ4010 IMPRT10 IMGW300 VEP12FF"
41  ndev=Words(graphics_device)

42  /*  Default graphics device is IMGW300  */

43  device=Find(graphics_device,'IMGW300')

44  /*  Parse the arguments  */

45  Arg fn ft fm . '(' options

46  If fn="" Then Do
47    Say ' '
48    Say 'You must enter a filename'
49    Say ' '
50    Exit 999
51    End
52  If ft="" Then ft="PDEVLIN"
53  If fm="" Then fm="*"
```

```
54  /*  Scan the options string  */

55  opt_err=false
56  Do While Words(options)>0

57    opt=Strip(Word(options,1),'B',' ') /* Strip leading/trailing blanks */
58    found=false

59    Do idev=1 to ndev  /*  Is this option a graphics device?  */
60      If Substr(opt,1,3)=Substr(Word(graphics_device,idev),1,3) Then Do
61        found=true
62        device=idev
63        Parse Var options . options  /*  Remove this opt from the list  */
64        End
65      End

66    If ^found Then Select

67    When Substr(opt,1,2)="XT" Then Do  /*  An X translation */
68      found=true
69      Parse Var options . xtrans options
70      If Datatype(xtrans)/="NUM" Then Do
71        opt_err=true
72        Say ' '
73        Say 'X translation must be a number, found "'xtrans'" instead'
74        End
75      End

76    When Substr(opt,1,2)="YT" Then Do  /*  A Y translation */
77      found=true
78      Parse Var options . ytrans options
79      If Datatype(ytrans)/="NUM" Then Do
80        opt_err=true
81        Say ' '
82        Say 'Y translation must be a number, found "'ytrans'" instead'
83        End
84      End

85    When Substr(opt,1,2)="XZ" Then Do  /*  An X zoom factor  */
86      found=true
87      Parse Var options . xzvalue options
88      If Datatype(xzvalue)/="NUM" Then Do
89        opt_err=true
90        Say ' '
91        Say 'X zoom value must be a number, found "'xzvalue'" instead'
92        End
93      End

94    When Substr(opt,1,2)="YZ" Then Do  /*  A Y zoom factor  */
95      found=true
96      Parse Var options . yzvalue options
97      If Datatype(yzvalue)/="NUM" Then Do
98        opt_err=true
```

```
99          Say ' '
100         Say 'Y zoom value must be a number, found "'yzvalue'" instead'
101         End
102       End

103     When Substr(opt,1,2)="Z" Then Do  /*  A global zoom factor  */
104       found=true
105       Parse Var options . zvalue options
106       If Datatype(zvalue)/="NUM" Then Do
107         opt_err=true
108         Say ' '
109         Say 'Zoom value must be a number, found "'zvalue'" instead'
110         End
111       End

112     /*  Set particle tracking flags.  F=omit, T=plot  */

113     When Substr(opt,1,2)="IQ" Then Do
114       found=true
115       Parse Var options . options
116       Do Until Datatype(chg)/="NUM"
117         Parse Var options chg options
118         If chg=-1 Then electron="F"
119         If chg= 0 Then photon  ="F"
120         If chg=+1 Then positron="F"
121         End
122       options=chg options
123       End

124     /*  Draw a border around the plot?  */

125     When Substr(opt,1,3)="BOX" Then Do
126       found=true
127       Parse Var options . options
128       box="T"
129       End
130     Otherwise Nop
131     End

132     If ^found Then Do
133       opt_err=true
134       Parse Var options unknown options
135       Say ' '
136       Say 'Unknown option "'unknown'"'
137       End

138     End

139   If opt_err Then Do
140     Say 'Stopping due to option specification error(s).'
141     Exit 998
142     End

143   /*  Combine zoom factors  */
```

```
144  xzvalue=xzvalue*zvalue
145  yzvalue=yzvalue*zvalue

146  /*  Check for existence of specified file  */

147  'ESTATE' fn ft fm
148  If rc<>0 Then Do
149    Say ' '
150    Say 'Unable to locate input graphics file "'fn ft fm'"'
151    Say ' '
152    Exit rc
153    End

154  /*  Erase the old version of the output file if it exists  */

155  g_device=Word(graphics_device,device)

156  'ESTATE' fn g_device fmtdisk
157  If rc=0 Then Do
158    'ERASE' fn g_device fmtdisk
159    End

160  /*  Compile and load the EGS4PL code  */
161  /*  But first copy EGS4PL MORTRAN to the fmtdisk */
162  'COPY EGS4PL MORTRAN * $EGS4PL MORTRAN3 ' fmtdisk ' (REP'

163  'FI 1 DISK MORNEW87 DATA * (RECFM F LRECL 80 BLOCK 80'
164  'FI 5 DISK $EGS4PL MORTRAN3 ' fmtdisk ' (RECFM   LRECL 80 BLOCK 80'
165  'FI 6 DISK $EGS4PL MORTLIST ' fmtdisk ' (RECFM F LRECL 133 BLOCK 133'
166  'FI 7 DISK $EGS4PL FORTRAN ' fmtdisk ' (RECFM F LRECL 80 BLOCK 80'

167  'MORNEW87'

168  If rc>0 Then Do
169    Say ' '
170    Say 'MORTRAN error, rc = 'rc
171    Say ' '
172    Exit rc
173    End

174  /*'EXEC FORT $EGS4PL '  */
175  'FORTVS2 $EGS4PL'
176  If rc>4 Then Do
177    Say ' '
178    Say 'FORTRAN error, rc = 'rc
179    Say ' '
180    Exit rc
181    End

182  'EXEC GLOBSAVE ALL'  /*  Save the user's global libraries  */

183  'GLOBAL TXTLIB  VSF2FORT UGOBJLIB'
184  'GLOBAL LOADLIB VSF2LOAD'
```

```
185  'LOAD $EGS4PL NUCLEUS' g_device '( NOAUTO NODUP'
186  If rc>0 Then Do
187    Say ' '
188    Say 'LOAD error, rc = 'rc
189    Say ' '
190    oldrc=rc
191    Signal Done
192    End

193  /*  Run it  */

194  'FILEDEF 5 TERMINAL'
195  'FILEDEF 6 TERMINAL ( RECFM FBA LRECL 133'
196  'FILEDEF 10 DISK' fn ft fm

197  /*  Select UGS output file  */

198  'FILEDEF' g_device 'DISK' fn g_device fmtdisk

199  Queue device xzvalue yzvalue xtrans ytrans electron photon positron box

200  'START'

201  oldrc=rc

202  Done:

203  'EXEC GLOBREST ALL'  /*  Restore the libraries  */

204  Exit oldrc
```

Code Listing of the EGS4PL_B.COM Command File for VAX/VMS

```
1   $!
2   $!  EGS4PL_B.COM -- Command file for building an EGS4PL.EXE file
3   $!                  in the current directory.
4   $!
5   $!  Ray F. Cowan and W. R. Nelson
6   $!
7   $!  2 May 1988
8   $!
9   $!  MOD RFC 890423.  LINK now contains IMPRT10 and TALARIS devices.
10  $!
11  $!  This command file requires:
12  $!
13  $!  (1) MORTRAN3.DAT -- Hex (binary) version of MORNEW77 macros.
14  $!  (2) MORNEW77.EXE -- Fortran 77 MORTRAN3 preprocessor.
15  $!
16  $!  The symbol 'UG' must point to the directory containing the
17  $!  Unified Graphics 77 system.
18  $!
19  $!  NOTE:  The LINK command below must be modified to include all
```

```
$!          graphics devices desired; these have the standard names
$!          used by UGS77.  If devices are added, they must also be
$!          added to the DEVSTR array in EGS4PL.MORTRAN, and to the DEVICE
$!          qualifier KEYWORD list in EGS4PL.CLD.
$!
$ SET VERIFY
$!
$!  Define a symbol pointing to the Unified Graphics directory.
$!
$ UG :== SLAC$UG:  !  Typical example:  UG :== DISK1:[UGSYS]
$!
$!  Run MORTRAN.
$!
$ ASSIGN/USER MORTRAN3.DAT    FOR001
$ ASSIGN/USER EGS4PL.MORTRAN  FOR005
$ ASSIGN/USER EGS4PL.MORTLIST FOR006
$ ASSIGN/USER EGS4PL.FOR      FOR007
$ RUN MORNEW77
$!
$!  Run FORTRAN.
$!
$ FOR EGS4PL
$!
$!  Link it.
$!
$ LINK EGS4PL,'UG'NUCLEUS+ -
       SEQ4010+IMPRT10+IMGN300+TALARIS+VEP12FF,'UG'OBJLIB/LIB
$!
$!  We are done.
$!
$ SET NOVERIFY
$ Exit
```

Code Listing of the EGS4PL.CLD Command Definition File for VAX/VMS

```
!
!  EGS4PL.CLD -- Command definition file for the verb EGS4PL which
!                runs the EGS4 pseudo-device to real-device graphics
!                post processor.  The command EGS4PL takes several
!                optional arguments which control zoom, translations,
!                type of tracks displayed, etc., in the output plots.
!
!  Ray F. Cowan and W. R. Nelson
!
!  2 May 1988
!
!
!  MOD RFC 890423.  Added graphic device selection via options con-
!                   sisting only of the device name, i.e., "/SEQ4010"
!                   is the same as "/DEVICE=SEQ4010" which already
!                   works.  Also added the Talaris laser printer as
!                   a device option.  Also added output file selection
!                   via "/OUTPUT=filename".
```

```
!
DEFINE VERB EGS4PL
!
!  NOTE:  The IMAGE command should be modified to point to the directory
!           where the EGS4PL.EXE file exists.
!
        IMAGE "$USR:[RAY.EGS4]EGS4PL.EXE"
!
        PARAMETER P1, LABEL=PDEVFILE, PROMPT="Pseudo-graphics file",
                      VALUE(REQUIRED,TYPE=$FILE)
        QUALIFIER DEVICE, VALUE(TYPE=UGDEVICE,DEFAULT=IMGW300)
        QUALIFIER SEQ4010
        QUALIFIER IMPRT10
        QUALIFIER IMGW300
        QUALIFIER TALARIS
        QUALIFIER VEP12FF
        QUALIFIER OUTPUT, VALUE(TYPE=$FILE)
        QUALIFIER IQOFF,  VALUE(REQUIRED,TYPE=$NUMBER,LIST)
        QUALIFIER XZOOM,  VALUE(DEFAULT=1.0)
        QUALIFIER YZOOM,  VALUE(DEFAULT=1.0)
        QUALIFIER ZOOM,   VALUE(DEFAULT=1.0)
        QUALIFIER XTRANS, VALUE(DEFAULT=0.0)
        QUALIFIER YTRANS, VALUE(DEFAULT=0.0)
        QUALIFIER BOX
!
        DISALLOW ZOOM AND (XZOOM OR YZOOM)
!
!  Define keywords for Unified Graphics device types.
!
!  NOTE:  All Unified Graphics devices to be used by EGS4PL should be
!         added as separate keywords to the following list, and also to
!         the DEVSTR array in EGS4PL.MORTRAN.
!
DEFINE TYPE UGDEVICE
        KEYWORD SEQ4010 !  Sequential 4010 plot commands.
        KEYWORD IMPRT10 !  Imagen imPRESS language, 240 pixels/inch.
        KEYWORD IMGW300 !  Imagen imPRESS language, 300 pixels/inch.
        KEYWORD TALARIS !  Talaris laser printer.
        KEYWORD VEP12FF !  Versatec plotter, fan-fold paper.
!
```

Code Listing of the EGS4PL.MORTRAN Post-processor

This code works on both VM/SP and VAX/VMS via the use of the code-generation switch $SELECT_MACHINE. Two calls to this Mortran macro occur on lines 48 and 49 near the beginning of the file; one or the other should be commented out as appropriate for the machine at hand.

```
%L
"*********************************************************************"
"*****************************     STANFORD LINEAR ACCELERATOR CENTER"
"**      E G S 4 P L      **                                         "
"*****************************                    30 APR 1989/1200"
```

```
"*********************************************************************"
"  Copyright (C) 1987 by the Board of Trustees of the Leland        "
"  Stanford Junior University.  All Rights Reserved.                "
"*********************************************************************"
"                                                                   "
"  EGS4PL MORTRAN -- Process 2D Graphics Data Written bu SHOWPL.    "
"                                                                   "
"  R. F. Cowan and W. R. Nelson                                     "
"                                                                   "
"  12 June 1987                                                     "
"                                                                   "
"  Revisions:                                                       "
"                                                                   "
"  RFC 880501.  Modified for use on a VAX.  Uses the Command        "
"               Definition utility to define an 'EGS4PL' command    "
"               which parses the command line for options (used in "
"               place of REXX, which is available on IBM machines)."
"                                                                   "
"  RFC 890423.  Added several things:  (1) Talaris laser printer.   "
"               (2) New command line options for graphics device    "
"               selection.  Can now say simply '/SEQ4010' instead   "
"               of '/DEVICE=SEQ4010' (the old way still works too)."
"               (3) Changed default name of the output file to be   "
"               the name of the input file with extension = the     "
"               name of the selected graphics device.               "
"                                                                   "
"==================================================================="
"Macros to allow conditional generation of machine-dependent code. "

!NEWCONDITIONAL;

SET <ENDVAX>=ENDGENERATE;
SET <ENDIBM>=ENDGENERATE;

REPLACE {$SELECT_MACHINE {ARB;};}
WITH    {
        [IF]'{P1}'='VAX'
           [SET <GENIBM> = NOGENERATE;
            SET <GENVAX> = GENERATE;];

        [IF]'{P1}'='IBM'
           [SET <GENIBM> = GENERATE;
            SET <GENVAX> = NOGENERATE;];
        };


"==================================================================="
"Select the VAX or VM version to be generated (VAX or IBM).        "

"$SELECT_MACHINE VAX;"  "  Remove quotes for VAX version  "
 $SELECT_MACHINE IBM;   "  Remove quotes for IBM version  "

REPLACE {PARAMETER #=#;} WITH
   { REPLACE {{P1}} WITH {{P2}}}
```

```
52  PARAMETER $SEGSIZE=5000;  "  Size of general graphic segment  "

53  <GENVAX>;

54  INTEGER STATUS,CLI$PRESENT,CLI$GET_VALUE;
55  INCLUDE '($SSDEF)';
56  EXTERNAL CLI$_COMMA;

57  CHARACTER*256 PDEV_FILE;  "  Input filename  "
58  CHARACTER*64  DEV_NAME;
59  CHARACTER*256 OUT_FILE;  "  Output filename  "
60  LOGICAL OUT_EXT;
61  CHARACTER*64  CHAR_BUF; INTEGER*4 LCB;
62  LOGICAL PDEV_EXT;

63  <ENDVAX>;

64  INTEGER*4 DATA(32);
65  REAL*4 XZOOM,YZOOM;  "  Zoom factors  "
66  REAL*4 XTRANS,YTRANS;  "  Translation factors  "
67  INTEGER IDEV2D;  "  2D device selection flag  "
68  COMMON/SEGM/SEGMNT($SEGSIZE);  "  The graphics segment  "

69  LOGICAL QBOX;  "  True = draw a border around the plotting area  "
70  LOGICAL QOMIT;
71  LOGICAL QELECT,QPHOTO,QPOSIT;  "  Particle tracking flags  "
72                                 "  True = plot it, false = omit it  "

73  INTEGER RED/2/,GREEN/3/,YELLOW/5/;  "  UGS77 color indices  "

74  CHARACTER*28 DEVSTR(5)/'SEQ4010,GENIL,DDNAME=SEQ4010',
75                         'IMPRT10,GENIL,DDNAME=IMPRT10',
76                         'IMGN300,GENIL,DDNAME=IMGN300',
77                         'TALARIS,GENIL,DDNAME=TALARIS',
78                         'VEP12FF,GENIL,DDNAME=VEP12FF'/;

79  CHARACTER*80 OPEN_STR;  "  This contains the option string for UGOPEN  "

80  CHARACTER*17 QUALTY/'         ,         '/;  "  Line structure info  "

81  CHARACTER*8 BRTNES(5)/'    VDIM',   "  Brightness options  "
82                        '     DIM',
83                        '  MEDIUM',
84                        '  BRIGHT',
85                        ' VBRIGHT'/;

86  "  Read in VAX or VM options  "

87  <GENVAX>;

88  "  Get the input file specification  "

89  STATUS=CLI$GET_VALUE('PDEVFILE',PDEV_FILE,LPD);
```

```
90  "  Was a file extension specified?  If so, save the '.' position for  "
91  "  for use in constructing the output file specification.              "

92  PDEV_EXT=.FALSE.;
93  IPD=LPD+1;

94  DO I=LPD,1,-1  [
95    IF(PDEV_FILE(I:I).EQ.'.') [ PDEV_EXT=.TRUE.; IPD=I; EXIT; ]
96    IF(PDEV_FILE(I:I).EQ.']') EXIT;
97    ]

98  "  Provide the default pseudo-device file extension if none was  "
99  "  specified.  "

100 IF(.NOT.PDEV_EXT) [
101   PDEV_FILE(LPD+1:LPD+9)='.PDEVLIN';
102   LPD=LPD+8;

103   ]

104 OPEN(UNIT=10,FILE=PDEV_FILE(1:LPD),FORM='UNFORMATTED',
105      ACCESS='SEQUENTIAL',STATUS='OLD');

106 "  Get the output device identifier  "

107 IF(CLI$PRESENT('SEQ4010')) [
108   IDEV2D=1;  "  The value assigned to IDEV2D must be the index of the  "
109              "  corresponding device name in the array DEVSTR.          "
110   ] ELSE IF (CLI$PRESENT('IMPRT10')) [
111   IDEV2D=2;
112   ] ELSE IF (CLI$PRESENT('IMGW300')) [
113   IOEV2D=3;
114   ] ELSE IF (CLI$PRESENT('TALARIS')) [
115   IDEV2D=4;
116   ] ELSE IF (CLI$PRESENT('VEP12FF')) [
117   IDEV2D=5;
118   ] ELSE IF(CLI$PRESENT('DEVICE')) [
119   STATUS=CLI$GET_VALUE('DEVICE',DEV_NAME,LDN);
120   IF(LDN.NE.7) [ IDEV2D=3; ]
121     ELSE [
122       DO IDV=1,5 [
123         IF(DEV_NAME(1:7).EQ.DEVSTR(IDV)(1:7)) [ IDEV2D=IDV; EXIT; ]
124         ]
125     ]
126   ] ELSE [
127     IDEV2D=3;  "  Default is IMGW300  "
128   ]

129 "  Get the output file information  "

130 IF(CLI$PRESENT('OUTPUT')) [

131   "  Get the input file specification  "
```

```
132    STATUS=CLI$GET_VALUE('OUTPUT',OUT_FILE,LOF);

133    "  Was a file extension specified?  "

134    OUT_EXT=.FALSE.;

135    DO I=LOF,1,-1  [
136      IF(OUT_FILE(I:I).EQ.';') [ OUT_EXT=.TRUE.; EXIT; ]
137      IF(OUT_FILE(I:I).EQ.'.') [ OUT_EXT=.TRUE.; EXIT; ]
138      IF(OUT_FILE(I:I).EQ.']') EXIT;
139      ]

140    "  Provide the default output file extension if none was  "
141    "  specified (the default is the name of the graphics device).  "

142    IF(.NOT.OUT_EXT) [
143      OUT_FILE(LOF+1:LOF+1)='.';
144      OUT_FILE(LOF+2:LOF+8)=DEVSTR(IDEV2D)(1:7);
145      LOF=LOF+8;
146      ]

147    ] ELSE [  "  OUTPUT option was not specified, so use defaults.  "

148    IF(PDEV_EXT) [
149      OUT_FILE(1:IPD)=PDEV_FILE(1:IPD);  "Everything but the extension"
150      OUT_FILE(IPD+1:IPD+7)=DEVSTR(IDEV2D)(1:7);
151      LOF=IPD+7;
152      ] ELSE [
153      OUT_FILE(1:IPD-1)=PDEV_FILE(1:IPD-1);  "  Up to, but not  "
154                                             "  including, the period  "
155      OUT_FILE(IPD:IPD)='.';
156      OUT_FILE(IPD+1:IPD+7)=DEVSTR(IDEV2D)(1:7);
157      LOF=IPD+7;
158      ]
159    ]

160  WRITE(6,'(''OPDEV_FILE='',/,'' '',A80)') PDEV_FILE(1:LPD);
161  WRITE(6,'(''OOUT_FILE='',/,'' '',A80)') OUT_FILE(1:LOF);

162  "  Get the zoom factors  "

163  XZOOM=1.0; YZOOM=1.0;  "  Default zoom factors.  Note that these  "
164                         "  will be overridden by the defaults in  "
165                         "  the EGS4PL.CLD file if the user speci-  "
166                         "  fies a qualifier like /XZOOM without an  "
167                         "  explicit value, e.g., /XZOOM=1.5  "
168                         "  This is true for all the other numeric  "
169                         "  options.  "

170  IF(CLI$PRESENT('ZOOM')) [
171    STATUS=CLI$GET_VALUE('ZOOM',CHAR_BUF,LCB);
172    READ(CHAR_BUF(1:LCB),*) ZOOM;  "  Convert from character string  "
173                                   "  to a REAL*4 number  "
```

```
174     XZOOM=ZOOM; YZOOM=ZOOM;
175     ]
176   ELSE [
177     IF(CLI$PRESENT('XZOOM')) [
178       STATUS=CLI$GET_VALUE('XZOOM',CHAR_BUF,LCB);
179       READ(CHAR_BUF(1:LCB),*) XZOOM;  "  Convert from character string  "
180       ]                               "  to a REAL*4 number             "
181     IF(CLI$PRESENT('YZOOM')) [
182       STATUS=CLI$GET_VALUE('XZOOM',CHAR_BUF,LCB);
183       READ(CHAR_BUF(1:LCB),*) YZOOM;  "  Convert from character string  "
184       ]                               "  to a REAL*4 number             "
185     ]

186   "  Get the translation values  "

187   XTRANS=0.0; YTRANS=0.0;  "  Defaults  "

188   IF(CLI$PRESENT('XTRANS')) [
189     STATUS=CLI$GET_VALUE('XTRANS',CHAR_BUF,LCB);
190     READ(CHAR_BUF(1:LCB),*) XTRANS;
191     ]

192   IF(CLI$PRESENT('YTRANS')) [
193     STATUS=CLI$GET_VALUE('YTRANS',CHAR_BUF,LCB);
194     READ(CHAR_BUF(1:LCB),*) YTRANS;
195     ]


196   "  Selectively turn off electrons, positrons, and photons  "

197   QELECT=.TRUE.; QPHOTO=.TRUE.; QPOSIT=.TRUE.;  "  Default (.TRUE.) is  "
198                                                 "  to draw them         "

199   IF(CLI$PRESENT('IQOFF')) [
200     :IQ_LOOP:
201     STATUS=CLI$GET_VALUE('IQOFF',CHAR_BUF,LCB);

202     READ(CHAR_BUF(1:LCB),*) IQOFF;
203     QELECT=QELECT.AND.IQOFF.NE.-1;
204     QPHOTO=QPHOTO.AND.IQOFF.NE. 0;
205     QPOSIT=QPOSIT.AND.IQOFF.NE.+1;

206     "  There may be a list of values separated by commas  "

207     IF(STATUS.EQ.%LOC(CLI$_COMMA)) GO TO :IQ_LOOP:;
208     ]

209   "  Draw a box?  "

210   QBOX=CLI$PRESENT('BOX');

211   "  Construct the VAX version of the UGOPEN options string, in  "
212   "  including the 'DDNAME=output file' specifier.  "
```

```
213  DO I=1,80 [ OPEN_STR(I:I)=' '; ]
214  OPEN_STR(1:7)=DEVSTR(IDEV2D)(1:7);
215  OPEN_STR(8:15)=',DDNAME=';
216  OPEN_STR(16:LOF+16)=OUT_FILE(1:LOF);

217  <ENDVAX>;
218  <GENIBM>;

219    "  Read the output device id, stacked by the REXX EXEC  "

220    READ(5,*) IDEV2D,XZOOM,YZOOM,XTRANS,YTRANS,QELECT,QPHOTO,QPOSIT,QBOX;

221  "  Construct the IBM version of the UGOPEN string, in particular not  "
222  "  including any output file specification, in contrast to the VAX.   "

223  DO I=1,80 [ OPEN_STR(I:I)=' '; ]
224  OPEN_STR=DEVSTR(IDEV2D);

225  <ENDIBM>;

226  OUTPUT XZOOM,YZOOM; ('0','Using X,Y ZOOM factors of ',G10.3,2X,G10.3);
227  OUTPUT XTRANS,YTRANS;
228  ('0','Using X,Y translations of ',G10.3,2X,G10.3);
229  OUTPUT; (' ','where -1=left or bottom, 0=center, +1=right or top');

230  :Loop_Over_Records:

231  "  Read an input record and process it  "


232  "READ(10,NUM=NBYTES,END=:Done:) DATA;   "
233  "READ(10,END=:Done:) DATA;"
234  READ(10,END=:Done:) DATA(1),(DATA(I),I=2,DATA(1));

235  "  We process the 6 types of records that may be supplied to this  "
236  "  routine  "

237  IF(DATA(2).EQ.1) [  "  Open record  "
238    CALL UGOPEN(OPEN_STR,2);
239    CALL UGDSPC('PUT',100000.,100000.,1.0);
240    ]
241  ELSEIF(DATA(2).EQ.2) [  "  Close record  "
242    CALL UGWRIT(' ',0,SEGMNT);
243    CALL UGCLOS(' ');
244    ]
245  ELSEIF(DATA(2).EQ.3) [  "  Beginning of picture record  "
246    CALL UGPICT('CLEAR',0);
247    CALL UGINIT('CLEAR',SEGMNT,$SEGSIZE);
248    IF(QBOX) [  "  Draw plotting area border  "
249      CALL UGLINE(' ',              0.0,     0.0,0,SEGMNT);
250      CALL UGLINE('SOLID,BRIGHT',   0.0,100000.0,1,SEGMNT);
251      CALL UGLINE('SOLID,BRIGHT',100000.0,100000.0,1,SEGMNT);
252      CALL UGLINE('SOLID,BRIGHT',100000.0,     0.0,1,SEGMNT);
253      CALL UGLINE('SOLID,BRIGHT',     0.0,     0.0,1,SEGMNT);
```

```
254       ]
255     ]
256   ELSEIF(DATA(2).EQ.4) [  "  Picture description record  "
257     "  Setup the line structure string  "
258     "  Select solid or dotted first:  photons (yellow) get dotted, all  "
259     "  else is solid  "
260     IF(DATA(4).EQ.YELLOW) [ QUALTY(1:8)='DOTTED  '; ]
261     ELSE                  [ QUALTY(1:8)='SOLID   '; ]
262     "  Select intensity level  "
263     QUALTY(10:17)=BRTNES(DATA(3));
264     "  Set the include/omit flag  "
265     QOMIT=(.NOT.QELECT .AND. DATA(4).EQ.GREEN ) .OR.
266           (.NOT.QPHOTO .AND. DATA(4).EQ.YELLOW) .OR.
267           (.NOT.QPOSIT .AND. DATA(4).EQ.RED   );
268     ]
269   ELSEIF(DATA(2).EQ.5) [  "  Mark data record  "
270   " OUTPUT DATA(1),DATA(2); ('0','Bad record type:  ',2I6);"
271     X=XZOOM*(FLOAT(DATA(3))-50000.0*(XTRANS+1.0))+50000.0;
272     Y=YZOOM*(FLOAT(DATA(4))-50000.0*(YTRANS+1.0))+50000.0;
273     CALL UGMARK('MARK=2',X,Y,SEGMNT);
274     ]
275   ELSEIF(DATA(2).EQ.6) [  "  Line end point record  "
276     X=XZOOM*(FLOAT(DATA(3))-50000.0*(XTRANS+1.0))+50000.0;
277     Y=YZOOM*(FLOAT(DATA(4))-50000.0*(YTRANS+1.0))+50000.0;
278     IF(.NOT.QOMIT) CALL UGLINE(QUALTY,X,Y,DATA(5),SEGMNT);
279     ]
280   ELSEIF(DATA(2).EQ.7) [  "  Character string record  "
281     OUTPUT DATA(1),DATA(2); ('0','Bad record type:  ',2I6);
282     STOP;
283     ]
284   ELSE [
285     OUTPUT DATA(2); ('0','Unknown record type encountered.',/,
286                      ' ','Type = ',I5,'  Stopping now.');
287     ]

288   GOTO :Loop_Over_Records:;.

289   :Done:

290   STOP;

291   END;
292   %E
293   SUBROUTINE UGXERR(LEVEL,NAME,INDEX);  "  Handle overflowing segments  "
294   INTEGER LEVEL,INDEX;
295   CHARACTER*8 NAME;
296   COMMON/SEGM/SEGMNT($SEGSIZE);

297   IF(INDEX.EQ.11) [
298     CALL UGWRIT(' ',0,SEGMNT);
299     CALL UGINIT('CONTINUE',SEGMNT,$SEGSIZE);
300     LEVEL=0;
301     ]

302   RETURN;
303   END;
304   %E
305   ;
```

REFERENCES

1. R. F. Cowan (MIT/LNS) and W. R. Nelson (SLAC), *Use of 3-D Color Graphics with EGS*, (abstract only), *Comput. Phys. Commun.* **45** (1987), p. 485.

2. Walter R. Nelson (SLAC), Hideo Hirayama (KEK, Tsukuba), and David W. O. Rogers (National Research Council, Ottawa), *The EGS4 Code Ssystem*, SLAC-0265, Dec 1985. 398pp.

3. Theodore M. Jenkins, Walter R. Nelson, Alessandro Rindi (editors), **Monte Carlo Transport of Electrons and Photons** (Plenum Press, N.Y., 1988).

4. J. D. Bjorken, S. Ecklund, W. R. Nelson, A. Abashian, C. Church, B. Lu, L. Mo, and T. Nunamaker, *Search for Neutral Metastable Penetrating Particles Produced in the SLAC Beam Dump*, Phys. Rev. **D38** (1988) 3375. [Also FERMILAB Print-88-0352 (1988)].

5. Robert C. Beach (SLAC), *The Unified Graphics System for Fortran 77: Programming Manual*, CGTM-203-Rev. 3, Oct 1988. 155pp. Includes the addendum to the UGS Programming manual. Revised version; and

   The Unified Graphics System for FORTRAN 77: Graphic Algorithms Manual, CGTM-204-Rev. 3, Oct 1988. 42pp. Revised version.

6. Robert C. Beach (SLAC), *The Unified Graphics System for Fortran 77: Internal Operation and Maintenance Manual*, CGTM-205, Oct 1988. 120pp. Includes the addendum to the UGS Maintenance Manual. Revised version.