(NASA-CR-185879) NETWORK, SYSTEM, AND STATUS SOFTWARE ENHANCEMENTS FOR THE AUTONOMOUSLY MANAGED ELECTRICAL POWER SYSTEM BREADBOARD. VOLUME 4: GRAPHICAL STATUS DISPLAY Research Report, Jun. 1988 - May

N91-10621

526841

Unclas

G3/61 0287454

The University of Alabama in Huntsville

JRC Research Report No. 90-18
June 1988 through May 1990

Network, System, and Status Software Enhancements for the Autonomously Managed Electrical Power System Breadboard

Graphical Status Display

Grant NAG8-720

Volume 4 of 4 Volumes

Prepared by

James W. McKee
University of Alabama, Huntsville
Huntsville, Alabama 35899
(205) 895-6257

Prepared for

Norma Whitehead
EB12
NASA/MSFC
Huntsville, Alabama

Table of Contents

Introduction 3

Operation 3

Figure 1 Top Level LASEPS Status Screen 5

Figure 2 SAS Status Screen 6

Figure 3 Battery Data Screen 7

Figure 4 AMPS Load Center 8

Figure 5 Power Management & Distribution System Screen . . 9

Figure 6 Load Center Screen 10

Appendix A Structured flow chart for **init_screen** 11

Appendix B Structured flow charts for **laseps** 12

Appendix C.1 Main program to create graphics displays 0.0 & 0.1 24

Appendix C.2 -- 0.2 Draw segment 2 29

Appendix C.3 -- 0.3 Draw segment 6 31

Appendix C.4 -- 0.4 Draw segment 9 34

Appendix C.5 -- 0.5 Draw segment 10 39

Appendix D.1 Main program to present the graphics displays . 41

Appendix D.2 -- 2.0 SAS(1) and 3.0 SAS(2) 43

Appendix D.3 -- 4.0 BAT(1) and 5.0 BAT(2) 45

Appendix D.4 -- 6.0 AMPS_LC 46

Appendix D.5 -- 7.0 P3.1 amd 8.0 P3.2 48

Appendix D.6 -- 9.0 PMAD 49

Appendix D.7 -- 10.0 SUMMARY_DATA 52

Appendix D.8 -- Load Center Data 53

Appendix D.9 -- Library functions 54

Appendix D.10 -- Include file 55

Introduction

This volume contains the description, structured flow charts, prints of the graphical displays, and source code to generate the displays for the AMPS graphical status system.

These graphical displays were designed by Mr. Yi of UAH and Mrs. Norma Whitehead of NASA. They were coded by Mr. Yi. At the time these displays were designed certain portions of the status system were not completely defined. Therefore, even though this code is correct and functional, additions will probably be made to it. This code should be used as a guide in creating any additional displays.

The function of these displays is to present to the manager of the AMPS system a graphical status display with "hot boxes" that allow the manager to get more detailed status on selected portions of the AMPS system. This program was developed on the SUN work station in C using the DI-3000 graphics language from Precision Visuals Inc. The graphical display is the Tektronix 4125P. The equipment is located in the AMPS laboratory at NASA/MSFC.

Operation

The development of the graphical displays is divided into two processes:

1. create the screen images and store them in files on the computer and
2. run the status program which uses the screen images.

The code in **init_screen** generates the screen images that are needed by the status program **laseps**. The generated screens are stored in the file "segsave.dat." It takes a while for the computer to generate the screens and once generated the screens do not change, only the data in the screens varies with time. This design allows the manager to have quick response when examining the status of the power system.

When **laseps** is run figure 1 is displayed on the Tektronix screen. The small rectangle in the middle of the 24 KW Load Center box is the mouse curser. Only the left button on the three button mouse is active. There are ten "hot boxes" on the top level LASEPS display:

1. SAS (75 KW),
2. SAS (30 KW),
3. 108 Cell 189 AHr Vented NiCd Bat (top box),
4. 108 Cell 189 AHr Vented NiCd Bat (bottom box),
5. 24 KW Load Center (AMPS),
6. P3 (top box),
7. P3 (bottom box),
8. SSM PMAD TEST BED,

9. Summary Data, and
10. END.

By moving the mouse into any one of the boxes and clicking the left mouse button, the manager can go to a more detailed status displays for the respective box.

If the user selects either of the two SAS boxes figure 2 will be displayed. This display give the relation of the space station to the earth eclipse or sun, the time until the next sunlight, the time until the next eclipse, and the solar array current. The only "hot box" is the return to main box which returns to figure 1.

If the user selects either of the two battery boxes figure 3 is displayed. This display shows the voltage at each cell of the battery. Any key returns to figure 1.

If the user selects the 24 KW Load Center box, figure 4 is displayed. This screen gives the user the voltage, current, switches positions, switches temperatures, and a relative temperature. There are two "hot boxes" which allow the user go get numerical data or return to figure 1.

The design of the display for the P3 boxes is still not complete. At the present time if these poxes are selected a dummy function is called.

If the user selects SSM/PMAD TEST BED box, figure 5 is displayed. The five load centers at the bottom of the display are selectable by the mouse. If any of the load centers are selected figure 6 is displayed. Figure 6 displays the voltages and currents at various location in the selected load center.

If the user selects the Summary Data box, data from the controllers and sensors will be displayed. The design for this screen is not complete.

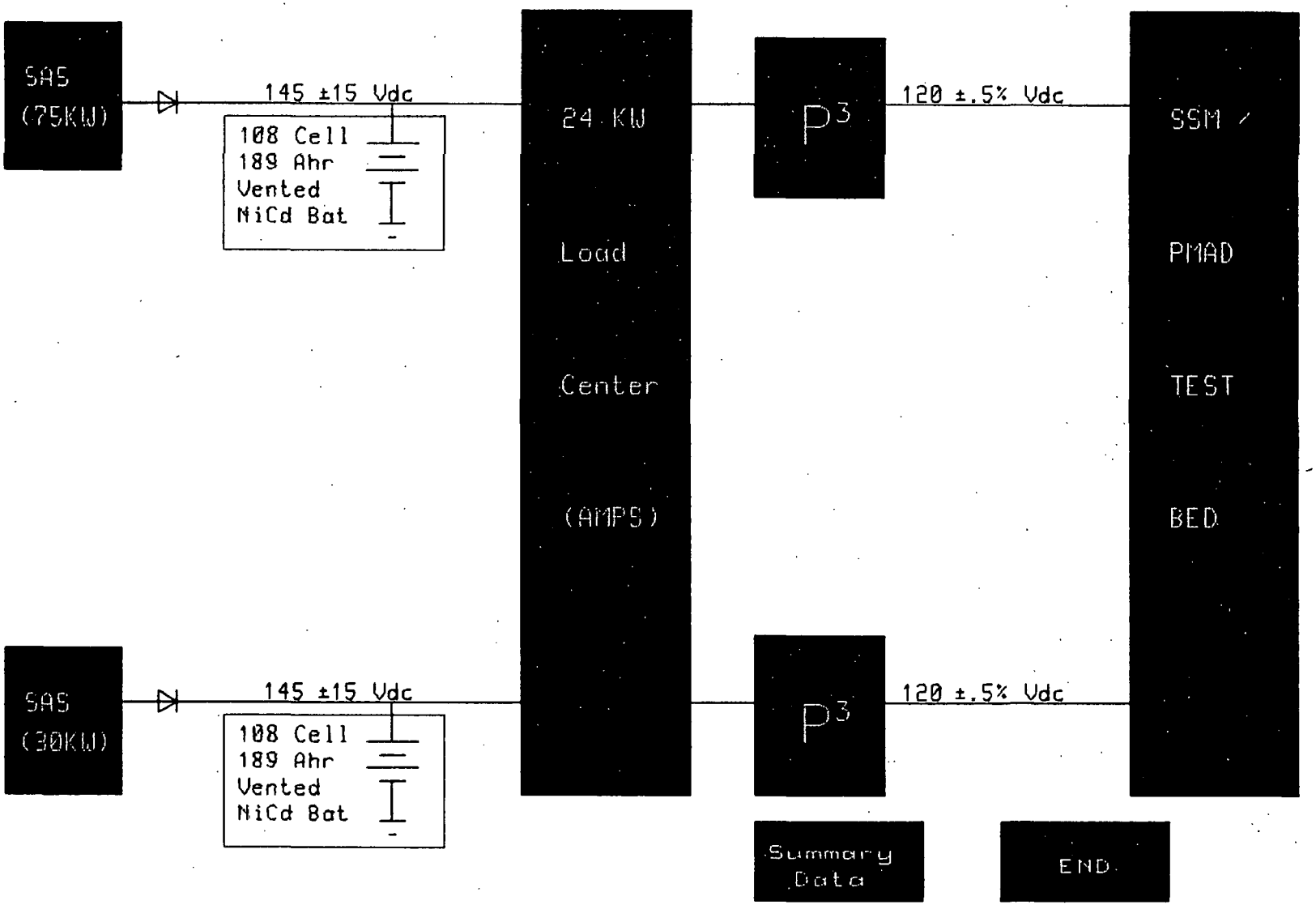


Figure 1 Top Level LASEPS Status Screen

LARGE AUTONOMOUS SPACECRAFT ELECTRICAL POWER SYSTEM
(LASEPS)

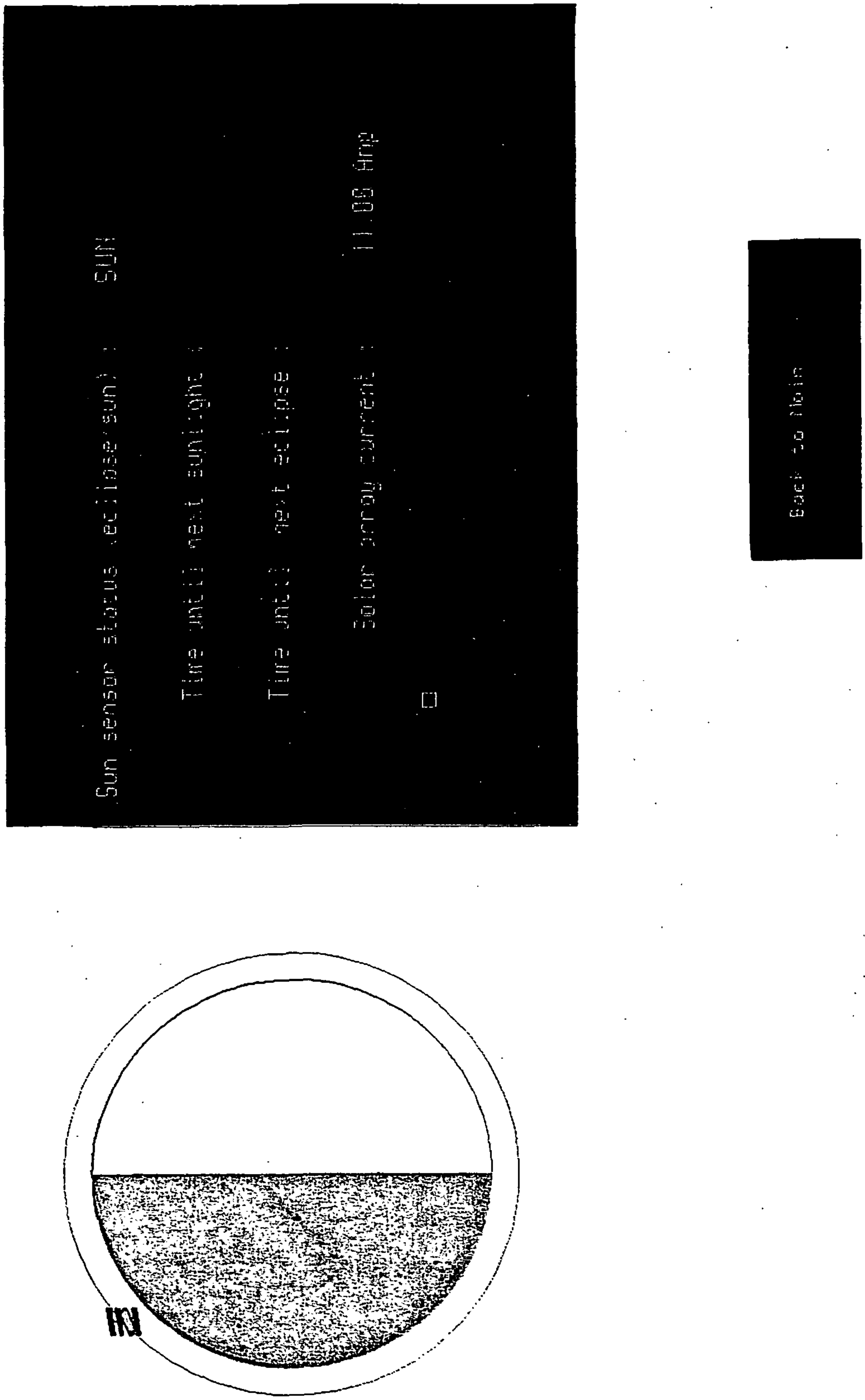


Figure 2 SAS Status Screen

```
********** BATTERY CELL DATA (Volts) ************************
 1 1.010 1.020 1.030 1.040 1.050 1.060 1.070 1.080 1.090 1.100 1.110 1.120
 2 1.020 1.030 1.040 1.050 1.060 1.070 1.080 1.090 1.100 1.110 1.120 1.130
 3 1.030 1.040 1.050 1.060 1.070 1.080 1.090 1.100 1.110 1.120 1.130 1.140
 4 1.040 1.050 1.060 1.070 1.080 1.090 1.100 1.110 1.120 1.130 1.140 1.150
 5 1.050 1.060 1.070 1.080 1.090 1.100 1.110 1.120 1.130 1.140 1.150 1.160
 6 1.060 1.070 1.080 1.090 1.100 1.110 1.120 1.130 1.140 1.150 1.160 1.170
 7 1.070 1.080 1.090 1.100 1.110 1.120 1.130 1.140 1.150 1.160 1.170 1.180
 8 1.080 1.090 1.100 1.110 1.120 1.130 1.140 1.150 1.160 1.170 1.180 1.190
 9 1.090 1.100 1.110 1.120 1.130 1.140 1.150 1.160 1.170 1.180 1.190 1.200
10 1.100 1.110 1.120 1.130 1.140 1.150 1.160 1.170 1.180 1.190 1.200 1.210
11 1.110 1.120 1.130 1.140 1.150 1.160 1.170 1.180 1.190 1.200 1.210 1.220
12 1.120 1.130 1.140 1.150 1.160 1.170 1.180 1.190 1.200 1.210 1.220 1.230
13 1.130 1.140 1.150 1.160 1.170 1.180 1.190 1.200 1.210 1.220 1.230 1.240
14 1.140 1.150 1.160 1.170 1.180 1.190 1.200 1.210 1.220 1.230 1.240 1.250
**** BATTERY CELL REFERENCE DATA ************************

PAUSE...HIT ANY KEY TO CONTINUE
```

Figure 3 Battery Data Screen

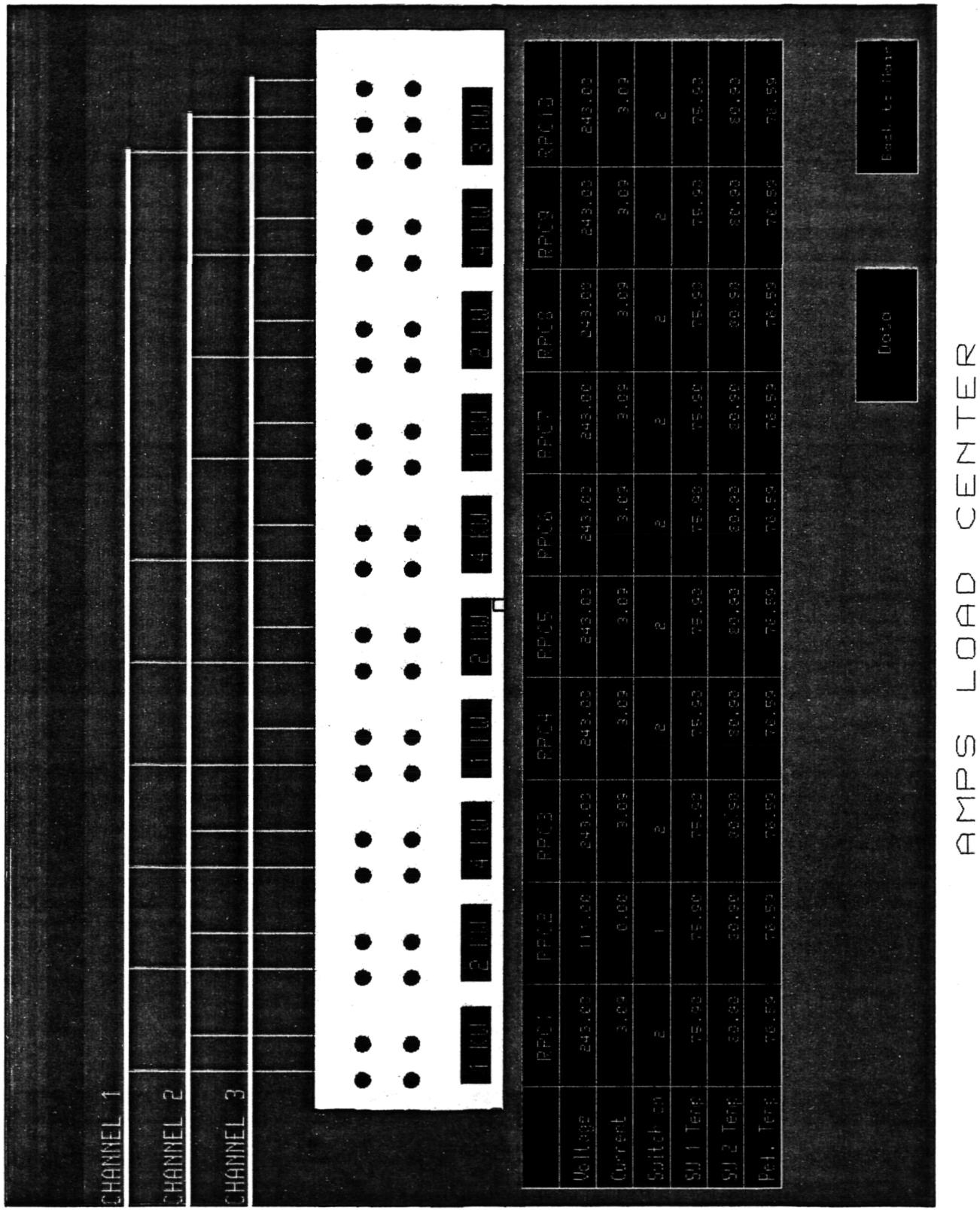


Figure 4 AMPS Load Center

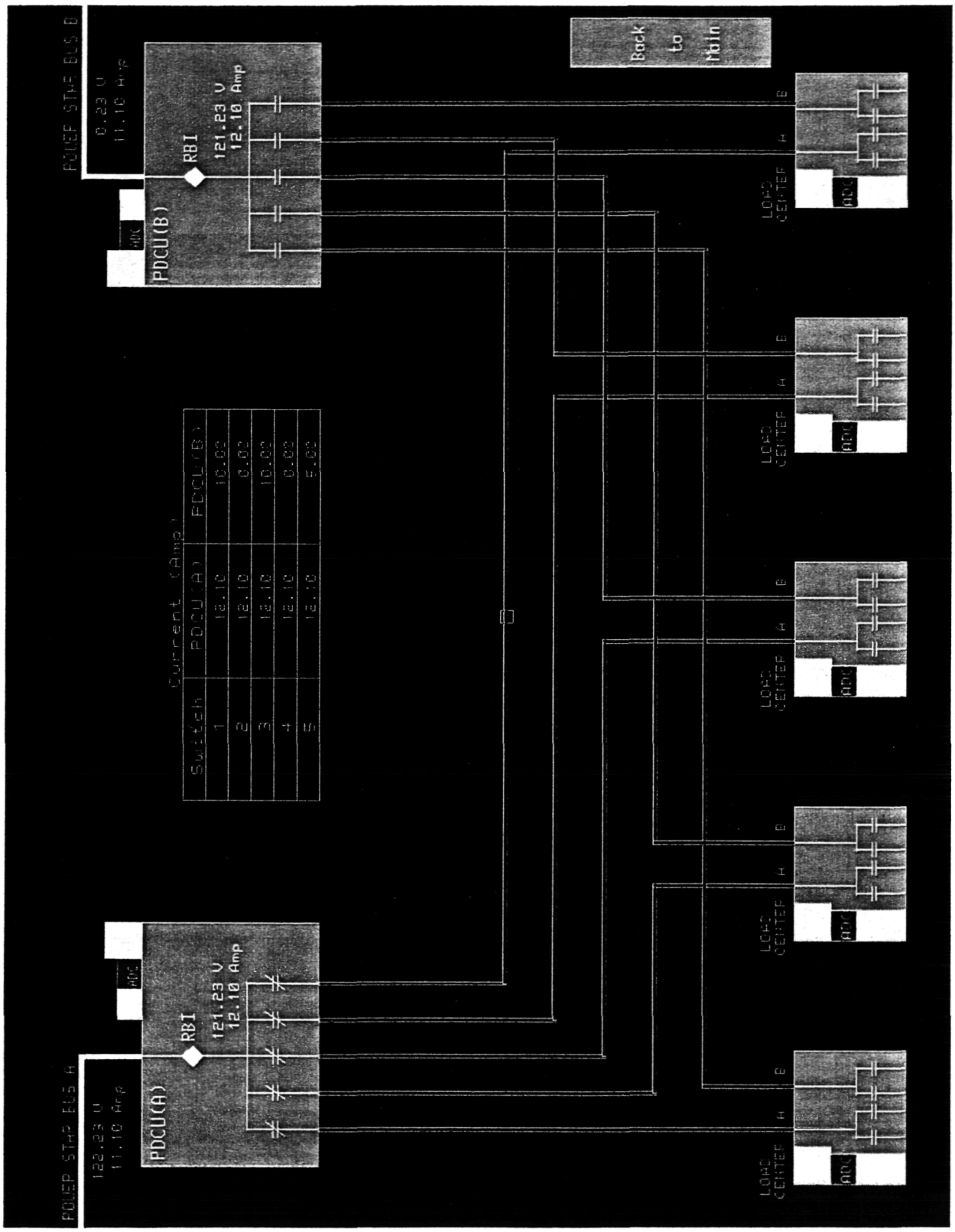


Figure 5 Power Management & Distribution System Screen

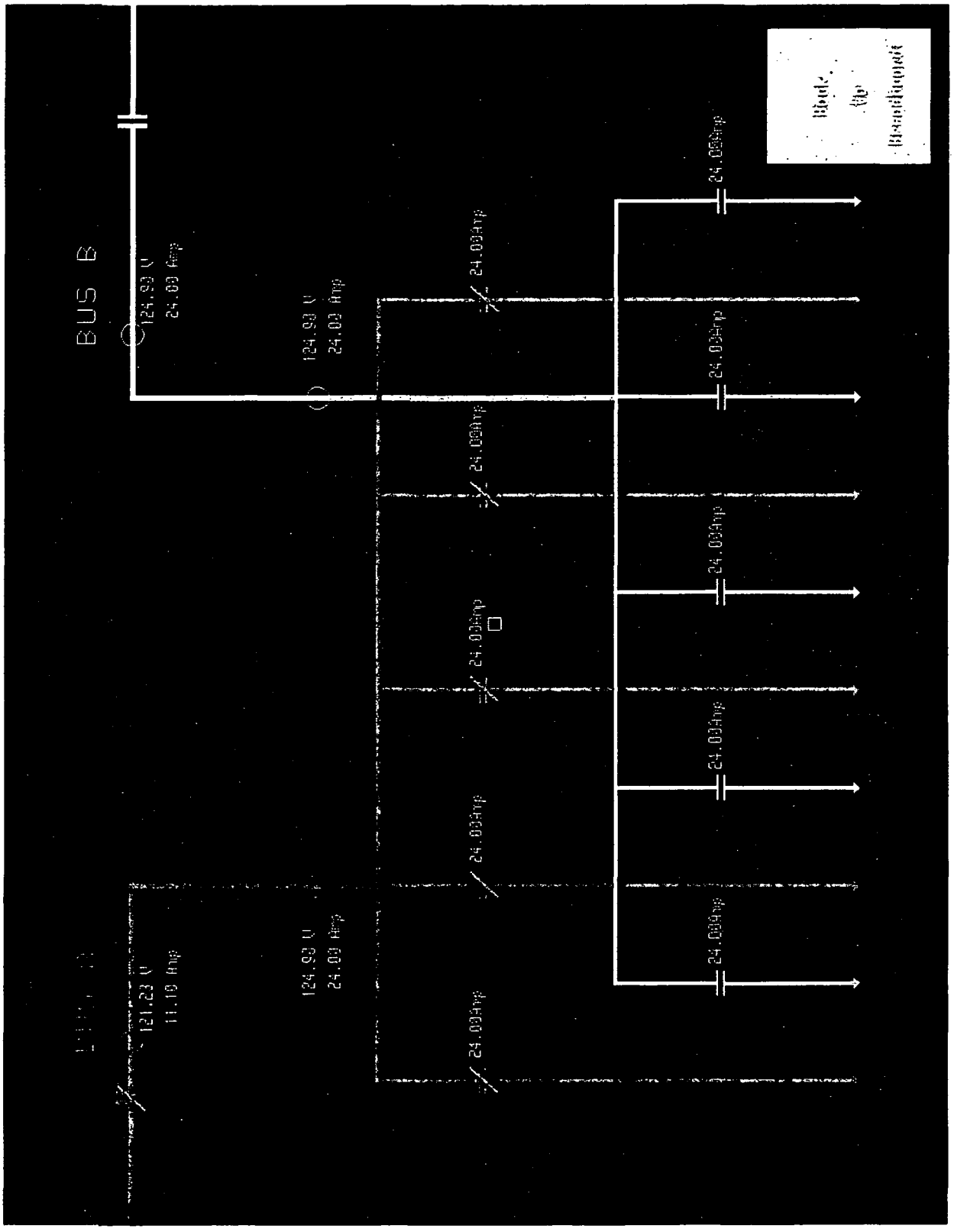
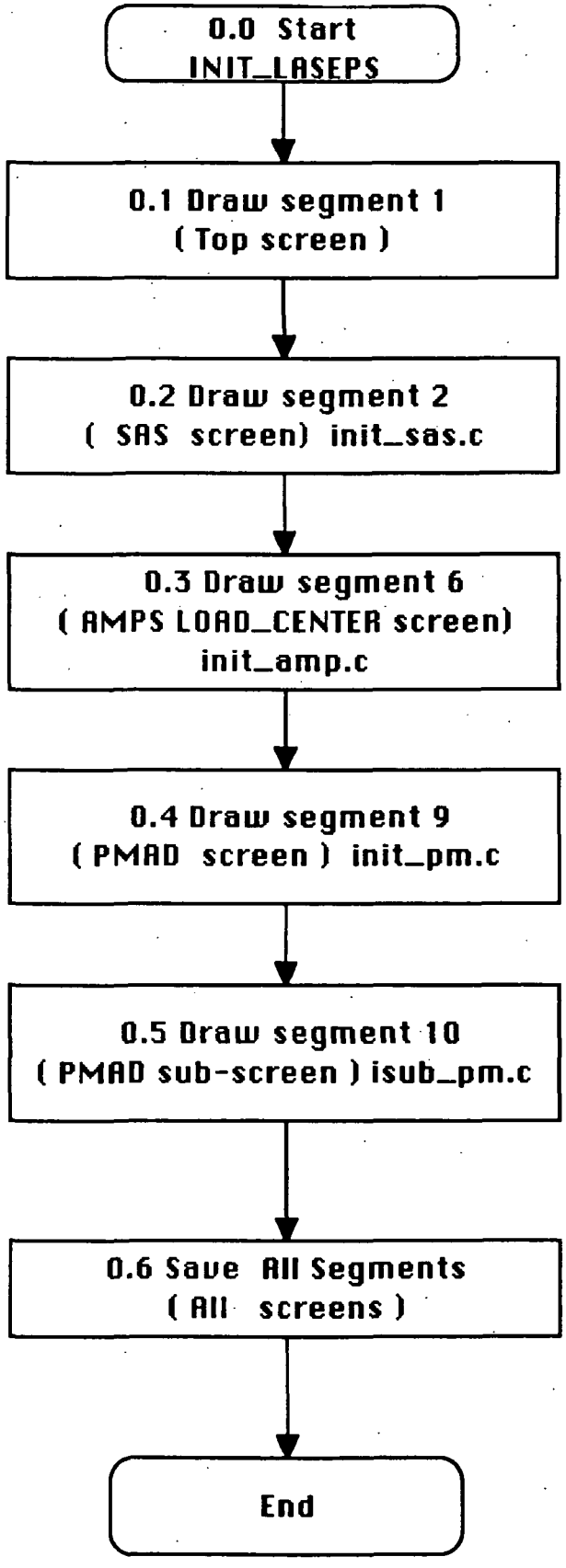
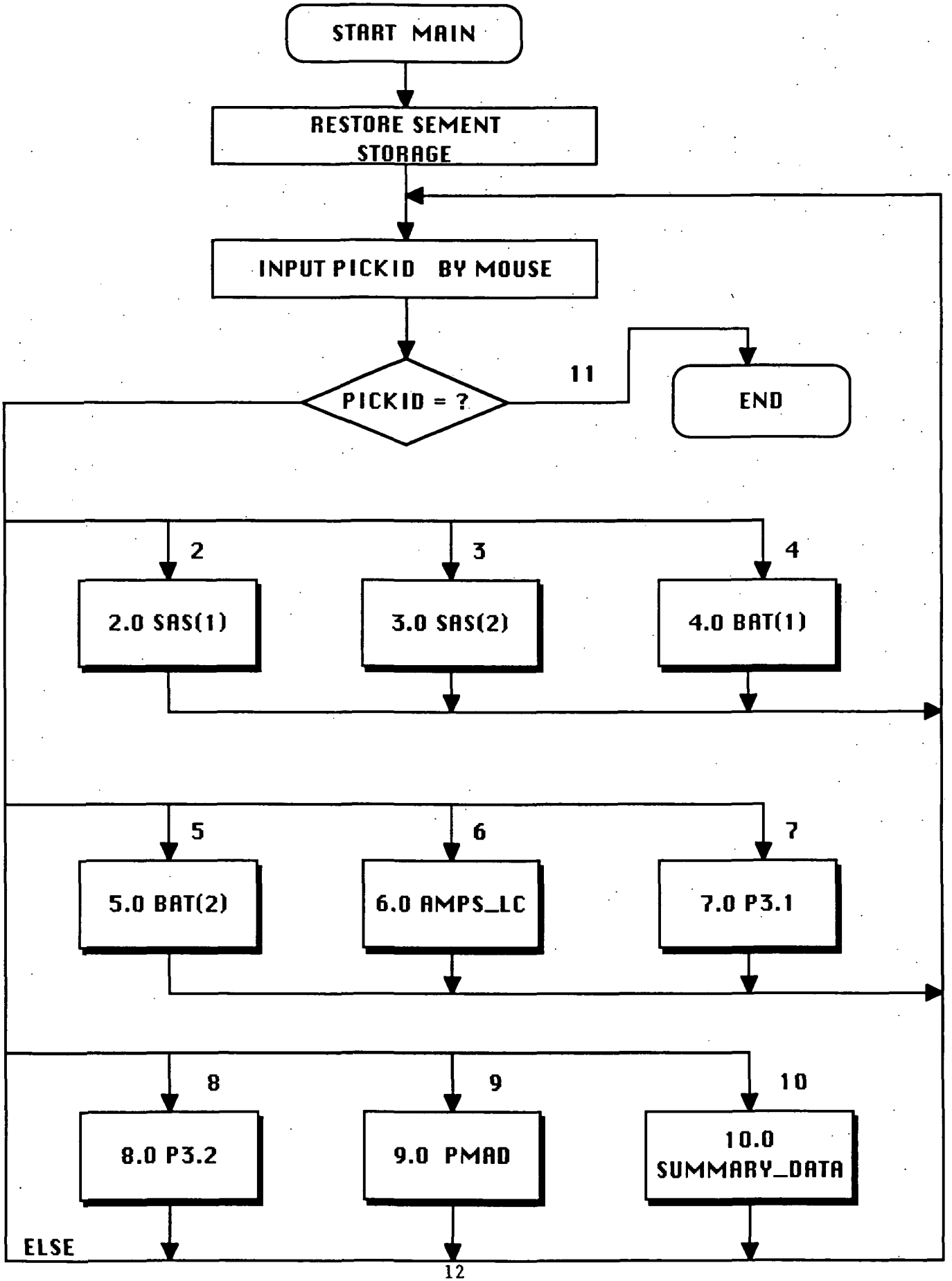
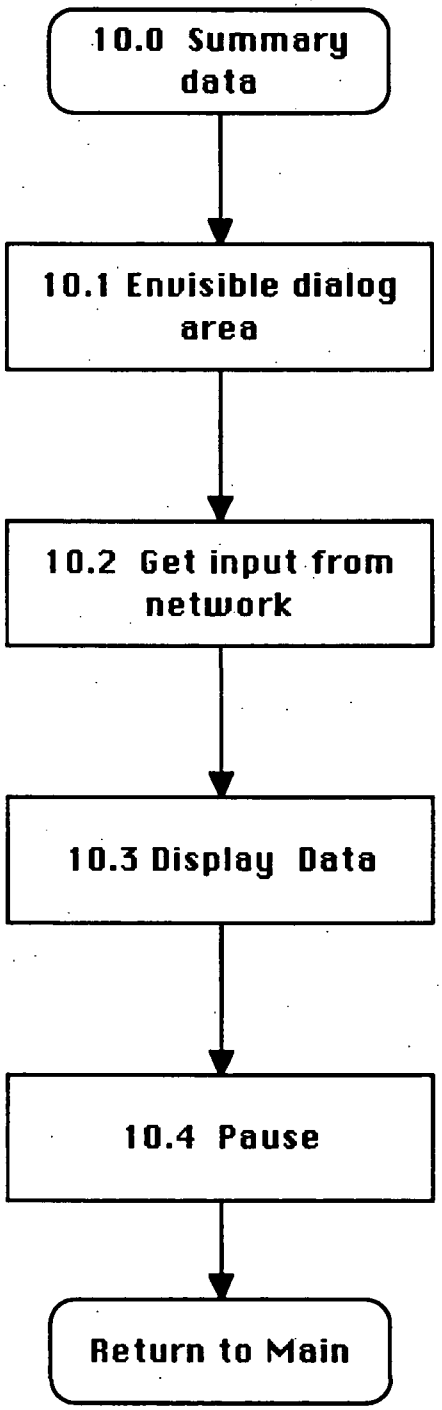


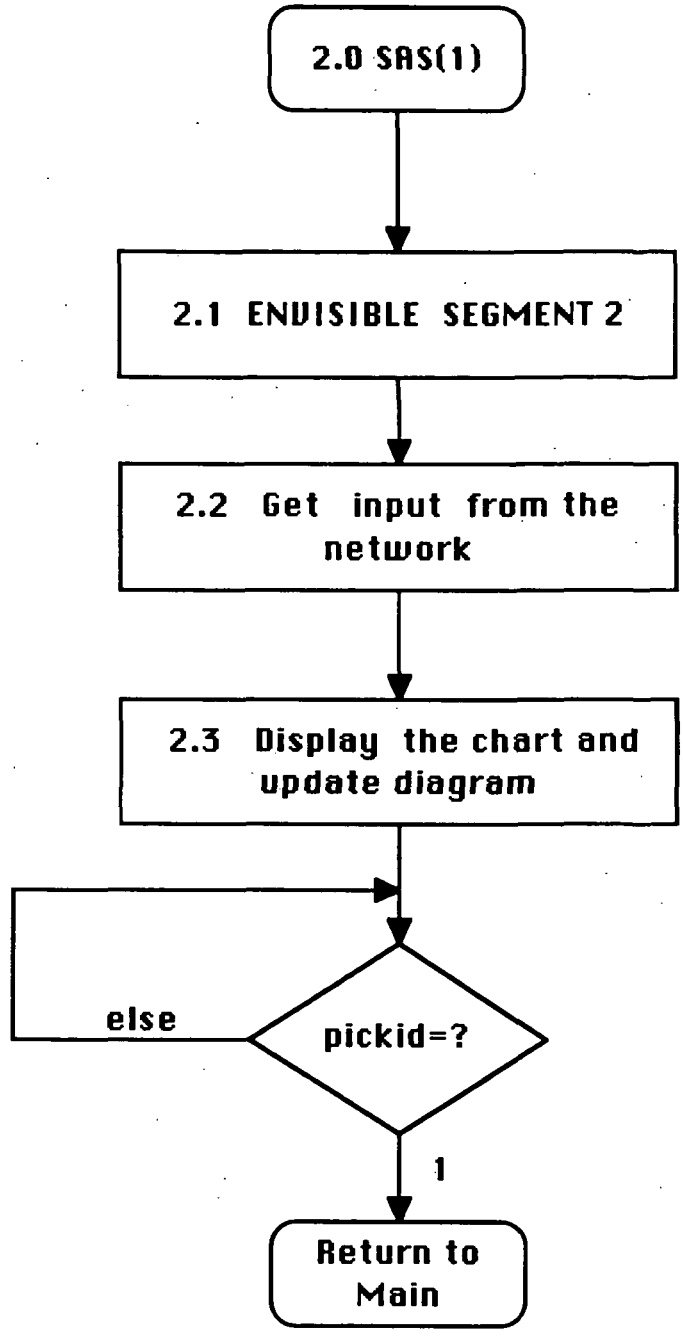
Figure 6 Load Center Screen

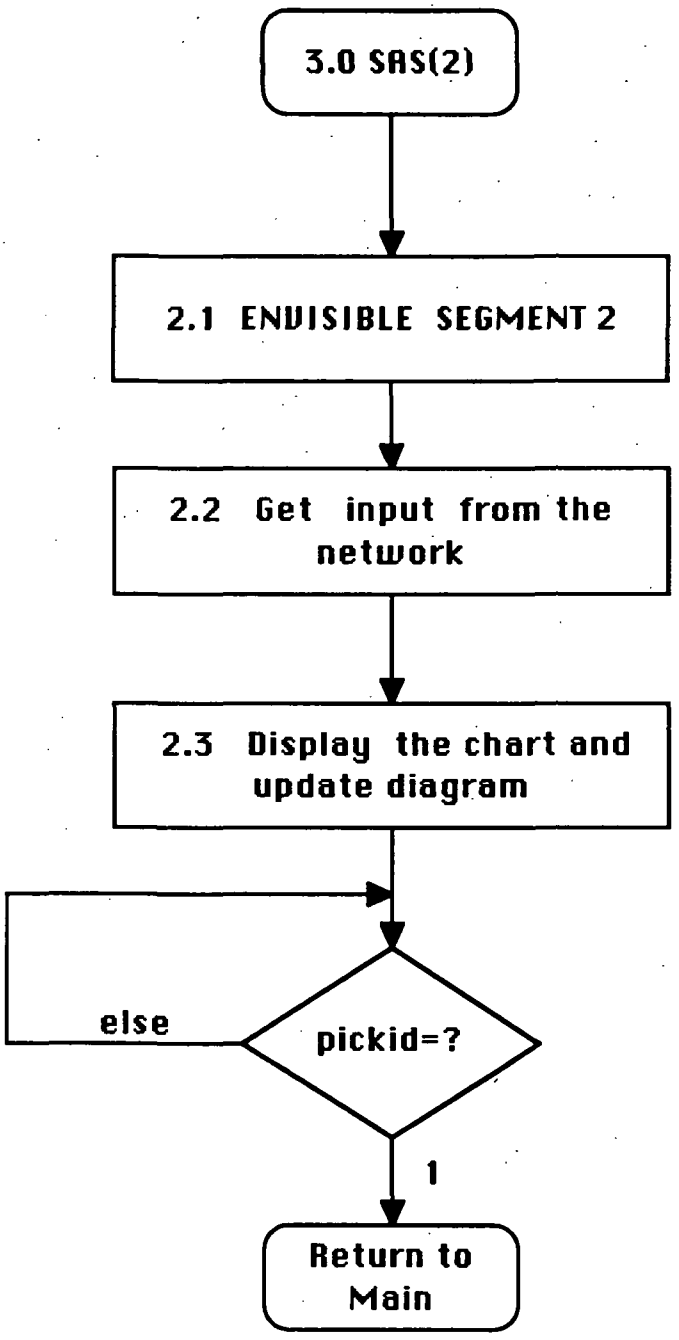
Appendix A Structured flow chart for init_screen

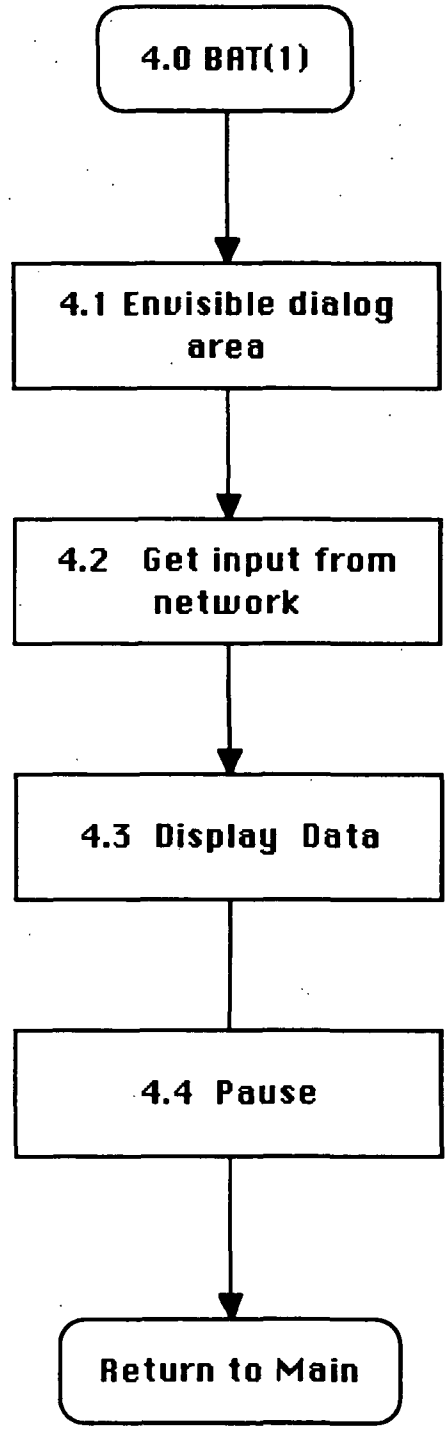


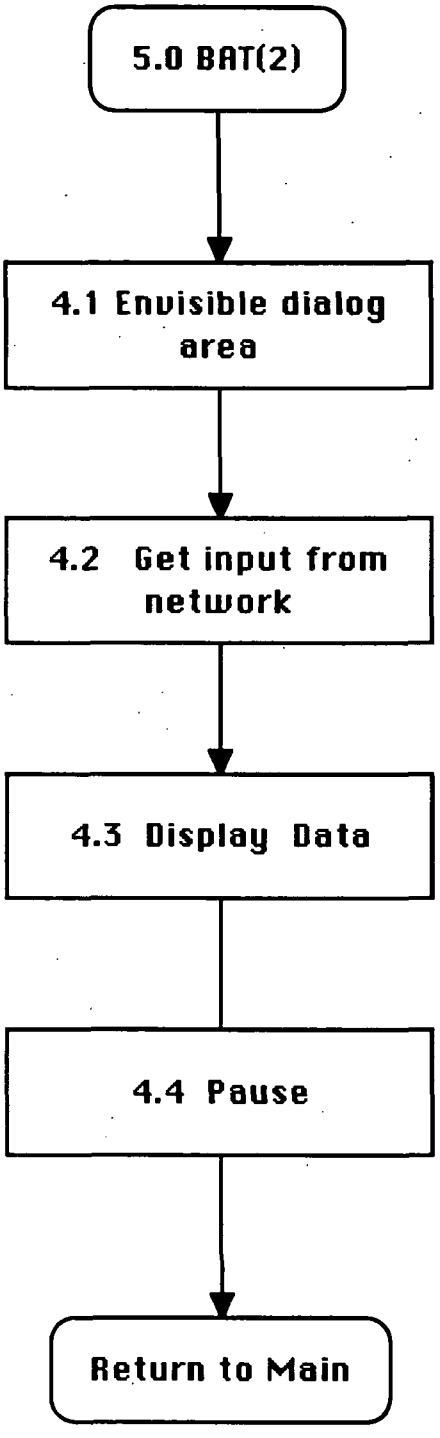

START MAIN
RESTORE SEMENT STORAGE
INPUT PICKID BY MOUSE
PICKID = ?
11
END
2
3
4
2.0 SAS(1)
3.0 SAS(2)
4.0 BAT(1)
5
6
7
5.0 BAT(2)
6.0 AMPS_LC
7.0 P3.1
8
9
10
8.0 P3.2
9.0 PMAD
10.0 SUMMARY_DATA
ELSE

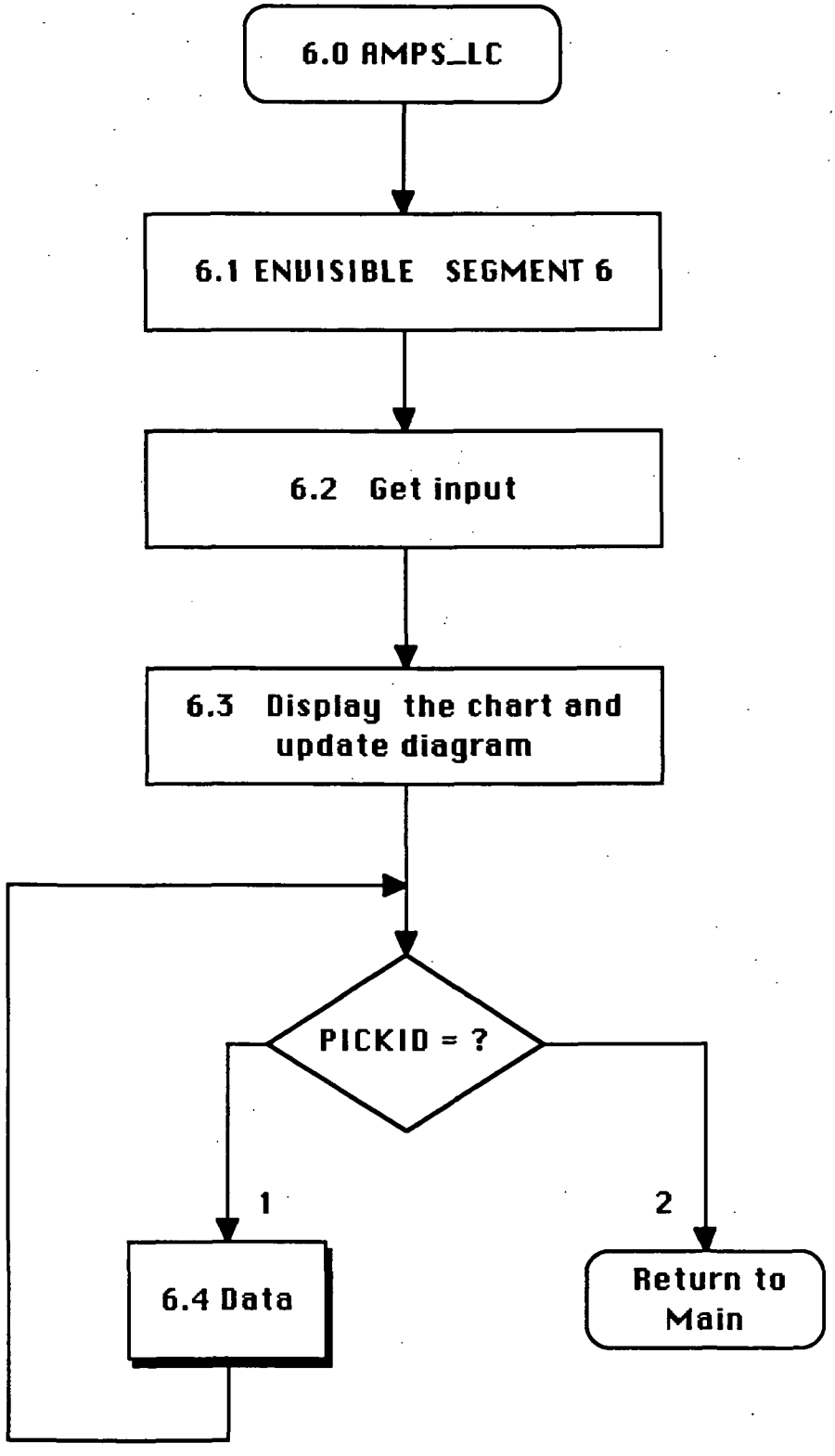

10.0 Summary data
10.1 Envisible dialog area
10.2 Get input from network
10.3 Display Data
10.4 Pause
Return to Main

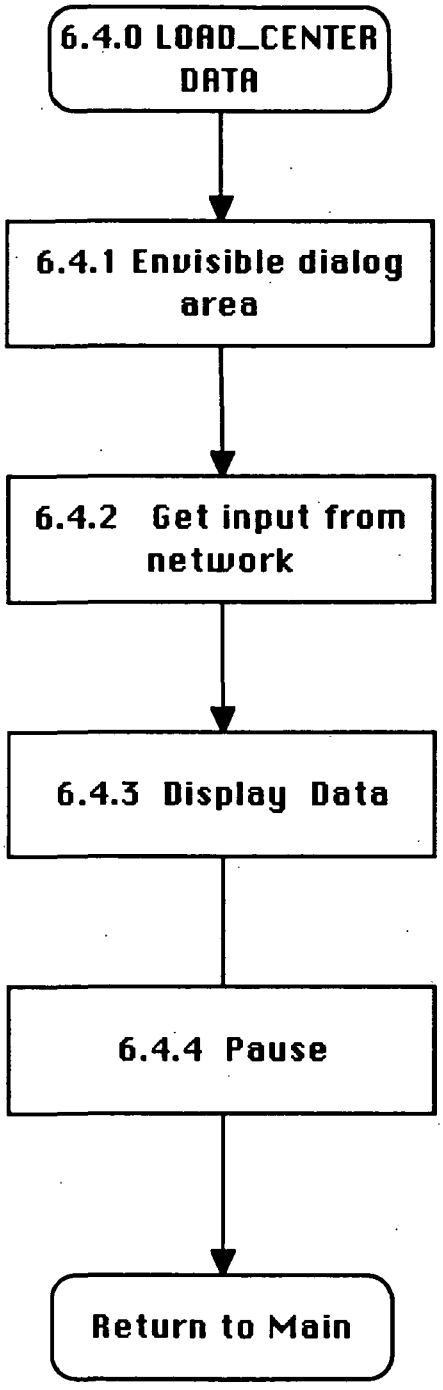

2.0 SAS(1)
2.1 ENVISIBLE SEGMENT 2
2.2 Get input from the network
2.3 Display the chart and update diagram
else
pickid=?
1
Return to Main


3.0 SAS(2)
2.1 ENVISIBLE SEGMENT 2
2.2 Get input from the network
2.3 Display the chart and update diagram
else
pickid=?
1
Return to Main

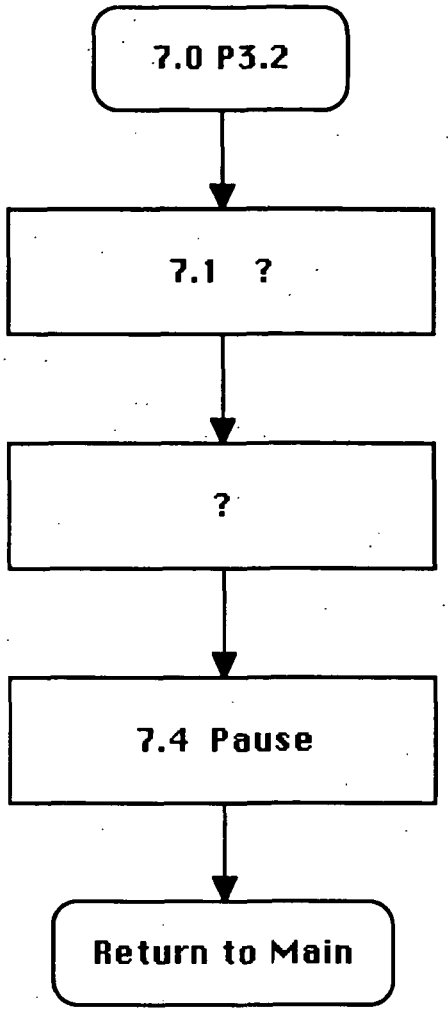

4.0 BAT(1)
4.1 Envisible dialog area
4.2 Get input from network
4.3 Display Data
4.4 Pause
Return to Main

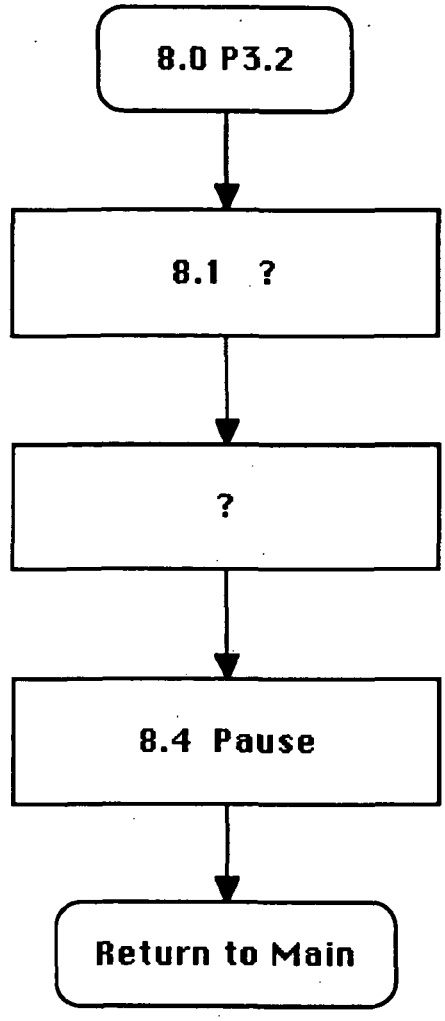

5.0 BAT(2)
4.1 Envisible dialog area
4.2 Get input from network
4.3 Display Data
4.4 Pause
Return to Main

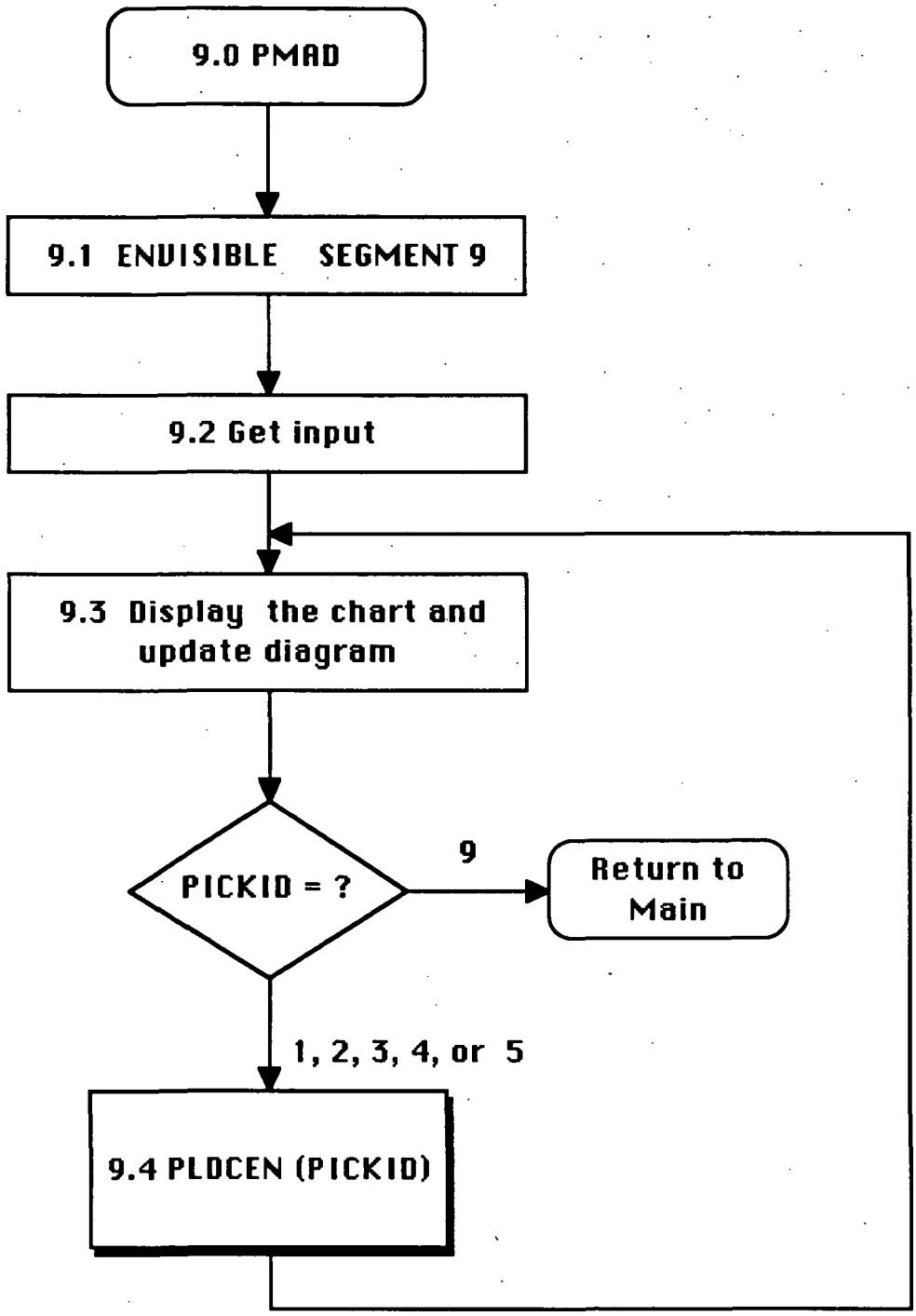

6.0 AMPS_LC
6.1 ENVISIBLE SEGMENT 6
6.2 Get input
6.3 Display the chart and update diagram
PICKID = ?
1
2
6.4 Data
Return to Main


6.4.0 LOAD_CENTER DATA
6.4.1 Envisible dialog area
6.4.2 Get input from network
6.4.3 Display Data
6.4.4 Pause
Return to Main

LASEPS/7.0 P3.1

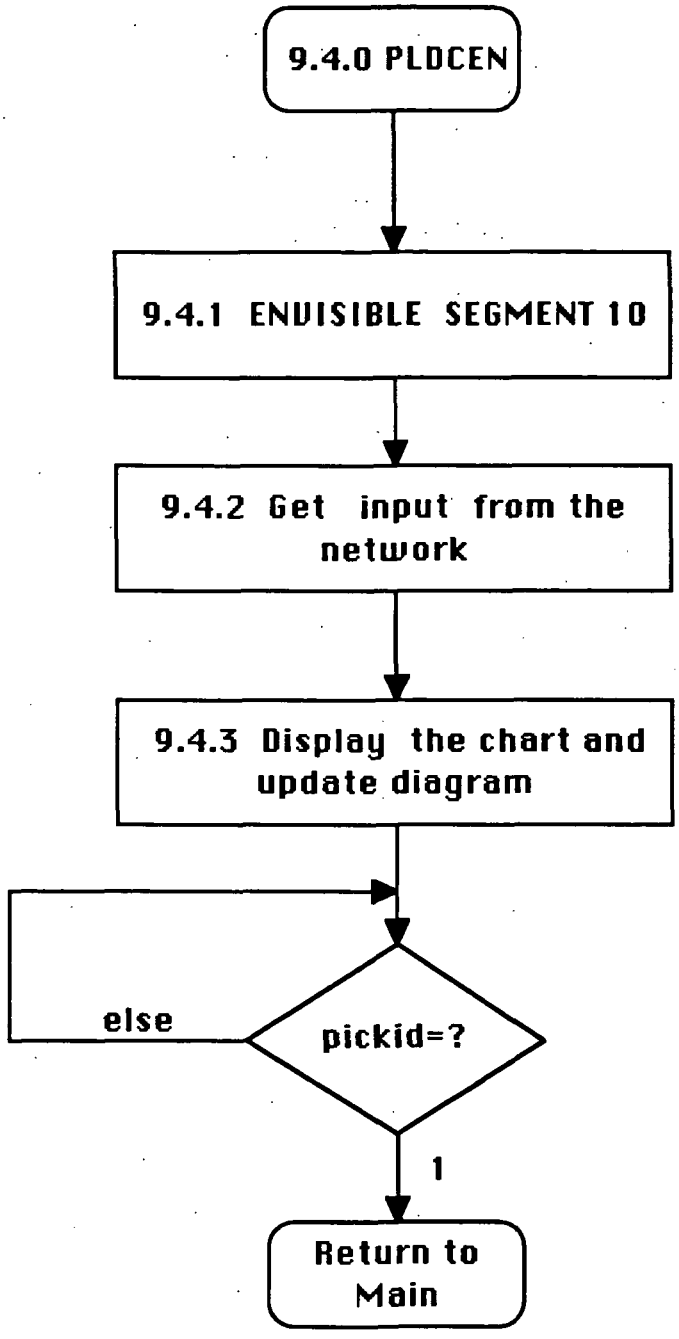



8.0 P3.2
8.1 ?
?
8.4 Pause
Return to Main


9.0 PMAD
9.1 ENVISIBLE SEGMENT 9
9.2 Get input
9.3 Display the chart and update diagram
PICKID = ?
9
Return to Main
1, 2, 3, 4, or 5
9.4 PLOCEN (PICKID)

ELSE

LASEPS/9.0 PMAD/9.4 PLOCEN (PICKID)



Appendix C.1 Main program to create graphics displays 0.0 & 0.1

```
/************** filename: init_laseps.c --- init_main program
                  made by Kwangsoo Yi    ***************/

#include "const.h"

main ()
{
float ratio, list[4];
int flag5=0, button, segnam, pickid, dumm1[5], dumm2[5];
     c_jbegin();
          c_jdinit(DSP);
          c_jdevon(DSP);
          c_jvport(-1.,1.,-.8,.8);
          c_jwindo(0.,100.,0.,80.);
          c_jdevwn(DSP,-1.,1.,-.8,.8);
/*     c_jfsopn(2, DSP, 9, "dbg");
          c_jsetdb(6);
 */    c_jescap(4111, 0, 0, dumm1, dumm2);/*dialog invisible*/
          c_jddete(1);
          c_jropen(1);           /* 0.1  Draw segment 1 */
               menu();
          c_jrclos();
          c_jvisbl(1, 0);
          c_jropen(2);           /* 0.2  Draw segment 2 */
               init_sas();
          c_jrclos();
          c_jvisbl(2, 0);
          c_jropen(6);           /* 0.3  Draw segment 6 */
               init_amp();
          c_jrclos();
          c_jvisbl(6, 0);
          c_jropen(9);           /* 0.4  Draw segment 9 */
               init_pm();
          c_jrclos();
          c_jropen(10);          /* 0.5  Draw segment 10 */
               isub_pm();
          c_jrclos();
          c_jpause(1);
          c_jescap(4110, 0, 0, dumm1, dumm2);/*dialog visible*/
          c_jssave();            /* 0.6  Save all Segments  */
          c_jdevof(DSP);
          c_jdend(DSP);
          c_jend();
}

menu()     /*  0.1  --- Top Screen   */
{
     c_jpintr(1); /*polygon fill*/
     c_jmove(3.,4.);
     c_jsize(1.82, 3.);
```

```
     c_j1strg("LARGE AUTONOMOUS SPACECRAFT ELECTRICAL POWER SYSTEM");
     c_jmove(40.,0.);
     c_j1strg("(LASEPS)");
     c_jcolor(GREEN);
     c_jmove(5.,25.);
     c_jdraw(95.,25.);
     c_jmove(5.,70.);
     c_jdraw(95.,70.);
     c_jpidex(CYAN ,10);
     c_jpkid(2);
          sas30(0.,18.);
     c_jpkid(3);
          sas75(0.,65.);
     c_jpkid(4);
          bat(10.,25.);
     c_jpkid(5);
          bat(10.,70.);
     c_jpkid(6);
          loadc(40.,18.);
     c_jpkid(7);
          pcube(58.,18.);
     c_jpkid(8);
          pcube(58.,63.);
     c_jpkid(9);
          pmad(87.,18.);
     c_jpkid(10);
          sdat(58.,10.);
     c_jpkid(11);
          end1(77.,10.);
}

sas30(x, y)
float x, y;
{
     c_jcolor(RED);
     c_jrect(x, y, x+9., y+11.);
   c_jrmove(1.,3.);
     c_jcolor(BLACK);
     c_jsize(1.2, 1.9);
     c_j1strg("(30KW)");
   c_jrmove(0.5,3.);
     c_j1strg("SAS");
}

sas75(x, y)
float x, y;
{
     c_jcolor(RED);
     c_jrect(x, y, x+9., y+11.);
   c_jrmove(1.,3.);
     c_jcolor(BLACK);
     c_j1strg("(75KW)");
```

```
   c_jrmove(0.5,3.);
     c_jlstrg("SAS");
}

bat(x, y)
float x, y;
{
     char str[9];
     c_jcolor(CYAN);
     c_jpidex(BLACK,10);
     c_jrect(x+7., y-11., x+24., y-1.);
   c_jrmove(1.,7.8);
     c_jcolor(0);
     c_jsize(1.1, 1.6);
     c_jlstrg("108 Cell");
   c_jrmove(0.,-2.);
     c_jlstrg("189 Ahr");
   c_jrmove(0., -2.);
     c_jlstrg("Vented");
   c_jrmove(0., -2.);
     c_jlstrg("NiCd Bat");
     c_jcolor(2);
     c_jmove(x, y-2.5);
     c_jsize(5., 5.); c_jfont(24); c_jhstrg("M"); c_jfont(1);
   c_jmove(x+20., y);
     c_jrdraw(0.,-3.);
     c_jrmove(-2., 0.);
     c_jrdraw(4.,0.);
     c_jrmove(-3.,-1.);
     c_jrdraw(2.,0.);
     c_jrmove(-3.,-1.);
     c_jrdraw(4.,0.);
     c_jrmove(-3.,-1.);
     c_jrdraw(2.,0.);
     c_jrmove(-1.,0.);
     c_jrdraw(0.,-3.);
     c_jrmove(-1., 0.);
     c_jrdraw(2.,0.);
     c_jrmove(-1.3,-1.);
     c_jrdraw(.6,0.);
   c_jmove(x+10., y);
     c_jsize(1.1, 1.6);
     c_jlstrg(i2a(145, str));
     c_jrmove(4.1,.0);plminus();
     c_jlstrg("15 Vdc");
}

loadc(x, y)
float x, y;
{
     c_jpidex(CYAN ,10);
     c_jcolor(RED);
```

```
      c_jrect(x, y, x+13., y+59.);
   c_jrmove(3.,50.);
      c_jcolor(BLACK);
      c_jsize(1.3, 1.9);
      c_jlstrg("24 KW");
   c_jrmove(0.,-10.);
      c_jlstrg("Load ");
   c_jrmove(0.,-10.);
      c_jlstrg("Center");
   c_jrmove(0.,-10.);
      c_jlstrg("(AMPS)");
}

pcube(x, y)
float x, y;
{
      char str[9];
      c_jcolor(RED);
      c_jrect(x, y, x+10., y+12.);
      c_jcolor(BLACK);
   c_jrmove( 3.3,3.);
      c_jsize(2.9, 4.4);
      c_jhstrg("P[BSUP]3[ESUP]");
   c_jrmove(8., 4.);
      c_jsize(1.1, 1.6);
      c_jcolor(GREEN);
      c_jlstrg(i2a(120, str)); c_jrmove(4., 0.);
   plminus(); c_jlstrg(".5% Vdc");
}
pmad(x, y)
float x, y;
{
      c_jcolor(RED);
      c_jrect(x, y, x+13., y+59.);
      c_jcolor(BLACK);
   c_jrmove(3.,50.);
      c_jsize(1.3, 1.9);
      c_jlstrg("SSM /");
   c_jrmove(0.,-10.);
      c_jlstrg("PMAD ");
   c_jrmove(0.,-10.);
      c_jlstrg("TEST ");
   c_jrmove(0.,-10.);
      c_jlstrg("BED  ");
}

sdat(x, y)
float x, y;
{
      c_jcolor(RED);
      c_jrect(x, y, x+13., y+6.);
      c_jcolor(BLACK);
```

```
    c_jrmove(3.,1.);
      c_jsize(1.4, 1.4);
      c_jlstrg("Data ");
    c_jrmove(-2.,2.);
      c_jlstrg("Summary ");
}

end1(x, y)
float x, y;
{
      c_jcolor(RED);
      c_jrect(x, y, x+13., y+6.);
      c_jcolor(BLACK);
    c_jrmove(3.,2.);
      c_jsize(1.4, 1.4);
      c_jlstrg(" END ");
}

plminus()
{
      c_jlstrg("+");
      c_jrmove(0.1, .4);
      c_jlstrg("_");
      c_jrmove(1., -.4);
}

/*******      end of file "init_laseps.c"    ********/
```

Appendix C.2 -- 0.2 Draw segment 2

```
/************* filename: init_sas.c --- 0.2 Draw segment 2
                    (SAS Screen)  **************/

#include "const.h"

init_sas ()
{
   static float rad, v, am, st, t;
   int i, j, button, segnam, pickid;
     char  str[20];
          rad=2.;
          orbit(rad, 16.,50.);
          sdata(41.,30.);
          stomain(60., 10.);
}

orbit(rad, x, y )
float rad, x, y ;
{
static float radius=14.;
   c_jcolor(GREEN); c_jpintr(1);
     c_jpidex(YELLO,100);
     c_jsectr(x, y, 0., radius, 0, -90., 90.);
     c_jpidex(GREY,100);
   c_jcolor(GREY);
     c_jsectr(x, y, 0., radius, 0, 270., 90.);
   c_jpintr(0); c_jcolor(GREEN);
     c_jcircl(x, y, 0., radius+1.9, 0.);
}

sdata(x, y)
float x, y;
{
     c_jpidex(WHITE,100); c_jpintr(1);
   c_jrect(x, y, x+59., y+40.);
     c_jrmove(2., 32.);
   c_jcolor(BLACK);
     c_jsize(1.0, 1.9);
     c_jlstrg("Sun sensor status (ECLIPSE/SUN) :");
     c_jrmove(0., -6.);
     c_jlstrg("       Time until next sunlight :");
     c_jrmove(0., -6.);
     c_jlstrg("       Time until  next eclipse :");
     c_jrmove(0., -6.);
     c_jlstrg("            Solar array current :");

}

stomain(x, y)
float x, y;
```

```
{
   c_jcolor(ORANG);
   c_jrect(x, y, x+23., y+8.);
     c_jrmove(3., 4.);
     c_jsize(0.92, 1.2);
   c_jcolor(BLACK);
     c_jlstrg("Back to Main");
}

/*******      End of file "init_sas.c"    ********/
```

Appendix C.3 -- 0.3 Draw segment 6

```
/************* filename: init_amp.c -- 0.3 Draw segment 6 **************/

#include "const.h"

init_amp ()
{
   static float v, am, st, t;
   static float volt[10][2], amp[10][2], swtemp[10][2], temp[10];
   static int onof, onoff[10][2], on123;
   int i, j, button, segnam, pickid;
     amenu();
     chart();
     c_jpkid(1);
          data(67., 6.);
     c_jpkid(2);
          ltomain(86., 6.);
}

amenu()
{
   c_jpintr(1);
   c_jpidex(CYAN , 100);
     c_jmove(29.,1.);
     c_jsize(2.72, 2.);
     c_j1strg("AMPS LOAD CENTER");
     c_jrect(0.0, 4.5, 100., 80.);
   c_jpidex(YELLO, 900);
     c_jrect(8.0, 39.5, 98., 55.);
   c_jpidex(BLACK, 100 );
     c_jcolor(BLACK);
     c_jrect(.0, 60., 94., 60.3);
     c_jsize(1.1, 2.);
     c_jrmove( 0.,.4); c_j1strg("CHANNEL 3");
     c_jrect(.0, 65., 91., 65.3);
     c_jrmove( 0.,.4); c_j1strg("CHANNEL 2");
     c_jrect(.0, 70., 88., 70.3);
     c_jrmove( 0.,.4); c_j1strg("CHANNEL 1");
     load(10.5, 47., "1", 3);
     load(19.0, 47., "2", 3);
     load(27.5, 47., "4", 3);
     load(36.0, 47., "1", 5);
     load(44.5, 47., "2", 5);
     load(53.0, 47., "4", 5);
     load(61.5, 47., "1", 6);
     load(70.0, 47., "2", 6);
     load(78.5, 47., "4", 6);
     load(87.0, 47., "3", 7);
}

load(x, y, kw, ch)
```

```
float x, y;
int ch; char kw[];
{
int flag=0;
float rad = .7;
   c_jpidex(WHITE, 19900);
     c_jcircl(x, y, 0., rad, 1);
     c_jcircl(x, y+4., 0., rad, 1);
     c_jcircl(x+3., y, 0., rad, 1);
     c_jcircl(x+3., y+4., 0., rad, 1);
     c_jmove(x+rad, y); c_jrdraw(.0, -2.);
     c_jrdraw(3., 0.);
     if (ch==7) {
        c_jcircl(x+6., y, 0., rad, 1);
        c_jcircl(x+6., y+4., 0., rad, 1);
        c_jrmove(rad, -4.); c_jrdraw(.0, -2.);
        c_jrdraw(-3., 0.);
          }
     c_jmove(x+3+rad, y); c_jrdraw(.0, -4.);
     c_jrmove(-4., -2.50);
     c_jrrect(6.5, 2.5);
     c_jrmove(.9, .0); c_j1strg(kw);
     c_jrmove( 2., .0);
     c_j1strg("KW");
     if (ch & 1) {
        c_jmove(x+rad, y+4.);
      c_jrdraw(0., 19.);
          flag++;
          }
     if (ch & 2) {
        c_jmove(x+rad+flag*3, y+4.);
      c_jrdraw(0., 14.);
          flag++;
          }
     if (ch & 4) {
        c_jmove(x+rad+flag*3, y+4.);
      c_jrdraw(0., 9.);
          flag++;
          }
}

chart()
{
   int i;
   c_jpidex(WHITE, 19900);
     c_jrect(1.5, 17., 97., 38.);
     c_jmove(10., 17.);
     c_jrdraw(0., 21.);
     for(i=0; i<6; i++) {
      c_jmove(1.5, 20.+i*3.);
          c_jrdraw(95.5, 0.);
          }
```

```
      for(i=1; i<10; i++) {
       c_jmove(8.5*i+10., 17.);
           c_jrdraw(0., 21.);
           }
      c_jmove(2.1, 17.4);
      c_jsize(0.8, 1.6);
      c_jlstrg("Ref. Temp");
      c_jrmove(0., 3.);
      c_jlstrg("SW 2 Temp");
      c_jrmove(0., 3.);
      c_jlstrg("SW 1 Temp");
      c_jrmove(0., 3.);
      c_jlstrg("Switch on");
      c_jrmove(0., 3.);
      c_jlstrg("Current");
      c_jrmove(0., 3.);
      c_jlstrg("Voltage");
      c_jrmove(10., 3.);
      c_jsize(1.048, 1.6);
      c_jlstrg("RPC1    RPC2    RPC3    RPC4    RPC5    RPC6    RPC7    RPC8
  RPC9    RPC10");
}

data(x, y)
float x, y;
{
   c_jrect(x, y, x+11., y+5.);
     c_jrmove(3., 2.);
     c_jsize(0.92, 1.3);
     c_jlstrg(" Data");
}

ltomain(x, y)
float x, y;
{
   c_jrect(x, y, x+11., y+5.);
     c_jrmove(1., 2.);
     c_jsize(0.82, 1.);
     c_jlstrg("Back to Main");
}

/*******     end of file "init_amp.c"   ********/
```

Appendix C.4 -- 0.4 Draw segment 9

```
/************* filename: init_pm.c --- 0.4 Draw segment 9
                (PMAD Screen)  **************/


#include "const.h"

init_pm ()
{
float ratio, list[4];
int flag, button, segnam, pickid, dumm1[5], dumm2[5];
     c_jpintr(1);
      c_jpidex(WHITE, 100);
        c_jmove(6.,1.);
        c_jsize(1.72, 2.);
        c_j1strg("POWER MANAGEMENT & DISTRIBUTION SYSTEM BREADBOARD ");
        c_jrect(0.0, 4.5, 100., 80.);
   c_jpidex(CYAN , 900);
     c_jcolor(BLACK);
     c_jpkid(1);
          pload(0);
     c_jpkid(2);
          pload(1);
     c_jpkid(3);
          pload(2);
     c_jpkid(4);
          pload(3);
     c_jpkid(5);
          pload(4);
     c_jpkid(9);
          p2main(92., 3.);
          pdcua(5., 55.);
          pdcub(77., 55.);
   pbus();
}

pload(n)
int n;
{               /* load center */
char str[10];
int i, d=20;
float x, y, offset = 3.5;
     x=n*d+offset; y=8.;
     c_jmove(x, y);
   c_jpidex(CYAN , 900);
   c_jrrect( 11., 9.);
     c_jpidex(YELLO,1);
   c_jrrect( 2.5, 4.);
     c_jsize(.8, 1.);
     c_jrmove(.1, .3);
     c_j1strg("SIC");
```

```
      c_jrmove(-0.1, 1.7);
      c_jrdraw(2.5, 0.);
      c_jrmove(-2.5, 0.0);
      c_jrmove(.1, .3);
      c_jlstrg("SIC");
      c_jrmove(-0.1, 1.7);
      c_jpidex(GREEN,1);
    c_jrrect( 2.5, 2.);
      c_jrmove(.1, .3);
      c_jlstrg("ADC");
      c_jrmove(-0.1, 1.7);
      c_jpidex(ORANG,1);
    c_jrrect( 3., 3.);
      c_jrmove(.3, .3);
      c_jlstrg("LLP");
      c_jrmove(-1.5, 3.3);
      c_jlstrg("CENTER");
      c_jrmove(6.5, 0.);
      c_jlstrg("A");
      c_jrmove(3.7, 0.);
      c_jlstrg("B");
      c_jrmove(-9.9, 1.2);
      c_jlstrg("LOAD");
    c_jmove(x+4.5, y+9.); c_jrdraw(0., -5.);
      c_jrmove(-.73, 0.); c_jrdraw(2.2, .0);
      c_jsize(1.34, 1.34); c_jfont(24);
      for (i=0; i<2; i++){
          c_jmove(x+3.9+2.1*i, y);
          c_jrdraw(0., 2.0);
          c_jrmove(.0, 0.8); c_jrdraw(0., 1.2);
          c_jrmove(-.67, -2.1);
          c_jhstrg("R");
            }
    c_jmove(x+8., y+9.); c_jrdraw(0., -5.);
      c_jrmove(-.61, 0.); c_jrdraw(2.2, .0);
      for (i=0; i<2; i++){
          c_jmove(x+7.5+2*i, y);
          c_jrdraw(0., 2.0);
          c_jrmove(.0, 0.8); c_jrdraw(0., 1.2);
          c_jrmove(-.67, -2.1);
          c_jhstrg("R");
            }
    c_jfont(1);
}

pdcub(x, y)
float x, y;
{
int i;
float dx[4], dy[4];
char str[10];
    c_jpidex(CYAN , 900);
```

```
   c_jrect( x, y, x+20., y+14.);
     c_jrmove(.5, 12.);
     c_jsize(.9, 1.5);
     c_j1strg("PDCU(B)");
     c_jmove(x+3., y+5.7); c_jrdraw(12.0, .0);
     c_jsize(1.5, 1.5); c_jfont(24);
     for (i=1; i<6; i++){
        c_jmove(x+3*i, y);
        c_jrdraw(0., 2.7);
        c_jrmove(.0, 1.66); c_jrdraw(0., 1.4);
        c_jrmove(-.75, -3.0);
        c_jhstrg("R");
         }
   c_jfont(1);
   c_jmove(x+9., y+5.7);
     c_jrdraw(0.0, 8.3);
     c_jrmove(0., -4.0);
   dx[0]=-.8; dx[1]=.8; dx[2]=.8; dx[3]=-.8;
   dy[0]= 0.; dy[1]=-.8; dy[2]=.8; dy[3]=.8;
   c_jpidex(BLACK, 900);
     c_jrplgn(dx,dy,4);
     c_jsize(.8, 1.4);
     c_jrmove(1.2, -0.3); c_j1strg("RBI");
   c_jrect(x+8.8, y+14., x+9.2, y+19.);
   c_jrect(x+9.2, y+19., x+23., y+18.6);
   c_jrmove(.3, 0.3); c_jsize(.8, 1.3);
     c_j1strg("POWER STAR BUS B");
     c_jmove(x, y+14.);
     c_jpidex(ORANG,1);
     c_jsize(.6, 1.);
   c_jrrect( 3., 3.);
     c_jrmove(.3, .3);
     c_j1strg("LLP");
     c_jrmove(2.7, -.3);
     c_jpidex(GREEN,1);
   c_jrrect( 2.5, 2.);
     c_jrmove(.2, 0.3);
     c_j1strg("ADC");
     c_jrmove(2.3, -.3);
     c_jpidex(YELLO,1);
   c_jrrect( 2.5, 2.);
     c_jrmove(.2, 0.3);
     c_j1strg("SIC");
}

pdcua(x, y)
float x, y;
{
int i;
float dx[4], dy[4];
char str[10];
   c_jpidex(CYAN , 900);
```

```
   c_jrect( x, y, x+20., y+14.);
     c_jrmove(.5, 12.);
     c_jsize(.9, 1.5);
     c_jlstrg("PDCU(A)");
     c_jmove(x+3., y+5.7); c_jrdraw(12.0, .0);
     c_jsize(1.5, 1.5); c_jfont(24);
     for (i=1; i<6; i++){
        c_jmove(x+3*i, y);
        c_jrdraw(0., 2.7);
        c_jrmove(.0, 1.66); c_jrdraw(0., 1.4);
        c_jrmove(-.75, -3.0);
        c_jhstrg("R");
          }
   c_jfont(1);
   c_jmove(x+9., y+5.7);
     c_jrdraw(0.0, 8.3);
     c_jrmove(0., -4.0);
   dx[0]=-.8; dx[1]=.8; dx[2]=.8; dx[3]=-.8;
   dy[0]= 0.; dy[1]=-.8; dy[2]=.8; dy[3]=.8;
   c_jpidex(BLACK, 900);
     c_jrplgn(dx,dy,4);
     c_jsize(.8, 1.4);
     c_jrmove(1.2, -0.3); c_jlstrg("RBI");
   c_jrect(x+8.8, y+14., x+9.2, y+19.);
   c_jrect(x-5.0, y+18.6, x+9.2, y+19.);
   c_jrmove(.6, .6); c_jsize(.8, 1.3);
     c_jlstrg("POWER STAR BUS A");
     c_jmove(x+12., y+14.);
     c_jpidex(YELLO,1);
   c_jrrect( 2.5, 2.);
     c_jsize(.6, 1.);
     c_jrmove(.2, 0.3);
     c_jlstrg("SIC");
     c_jrmove(2.3, -.3);
     c_jpidex(GREEN,1);
   c_jrrect( 2.5, 2.);
     c_jrmove(.2, 0.3);
     c_jlstrg("ADC");
     c_jrmove(2.3, -.3);
     c_jpidex(ORANG,1);
   c_jrrect( 3., 3.);
     c_jrmove(.2, 0.3);
     c_jlstrg("LLP");
}

pbus()
{
   int i;
     float abdist=3.5, width=.15;
   c_jpidex(BLUE , 900);
   c_jrect(8.-width, 17., 8.+width, 55.);
     for (i=1; i<5; i++) {
```

```
            c_jrect(i*3.+8.-width, i*4.+24., i*3.+8.+width, 55.);
            c_jrect(i*3.+8., i*4.+24., i*20.+8., i*4.+24.+2*width);
            c_jrect(i*20.+8.-width, 17., i*20.+8.+width, i*4.+24.+2*width);
        }
      for (i=1; i<5; i++) {
            c_jrect(i*3.+77.-width, i*4.+20., i*3.+77.+width, 55.);
            c_jrect(i*20.-8.5, i*4.+20., i*3.+77., i*4.+20.+2*width);
            c_jrect(i*20.-8.5-width, 17., i*20.-8.5+width, i*4.+20.+2*width);
        }
    c_jrect(92.-width, 17., 92.+width, 55.);
}

p2main(x, y)
float x, y;
{
    c_jpidex(CYAN , 900);
    c_jrect(95., 19., 99., 35.);
      c_jsize(0.72, 1.3);
      c_jrmove(.5, 4.);
      c_j1strg("Main");
      c_jrmove(0., 3.);
      c_j1strg(" to ");
      c_jrmove(0., 3.);
      c_j1strg("Back");
}
/******** end of file "init_pm.c"    *********/
```

Appendix C.5 -- 0.5 Draw segment 10

```
/************* filename: isub_pm.c --- 0.5 Draw segment 10
       (PMAD Sub-screen)  ***************\

#include "const.h"

isub_pm ()
{
float ratio, list[4];
int flag, button, segnam, pickid, dumm1[5], dumm2[5];
     c_jpintr(1);
      c_jpidex(WHITE, 100);
        c_jmove(16.,1.);
        c_jsize(1.72, 2.);
        c_jlstrg("  LOAD   CENTER");
        c_jrect(0.0, 4.5, 100., 80.);
     busa(28., 70.);
     busb(68., 70.);
   lbus(BLUE , 12., 50., 38.);
   lbus(BLACK , 20., 31., 19.);
     c_jpkid(1);
         back(92., 3.);
}

busb(x, y)
float x, y;
{
int i;
     float  width=.40;
     c_jpintr(1);
   c_jpidex(BLACK, 900);
   c_jcolor(BLACK);
     c_jmove(x+4., y+2.5);
     c_jlstrg("BUS B");
   c_jrect(x-width, y-width, 90., y);
     c_jrect(90., y-1.4, 90.25, y+1.0);
     c_jrect(90.75, y-1.4, 91.0, y+1.0);
   c_jrect(91.0, y-width, 100., y);
   c_jrect(x, y, x-width, y-39.);
     c_jpintr(0);
     c_jcircl(x+5., y-width/2, 0., 0.9, 1);
     c_jcircl(x-width/2, y-15.0, 0., 0.9, 1);
}

busa(x, y)
float x, y;
{
int i;
     float  width=.40;
     c_jpintr(1);
   c_jpidex(BLUE , 900);
```

```
   c_jcolor(BLUE );
     c_jmove(14., y+2.5);
     c_j1strg("BUS A");
   c_jrect(0., y-width, 10., y);
     c_jrect(10., y-1.4, 10.25, y+1.0);
     c_jrect(10.75, y-1.4, 11.0, y+1.0);
   c_jrect(11.0, y-width, x, y);
   c_jrect(x, y, x-width, y-20.);
     c_jpintr(0);
     c_jcircl(x-13., y-width/2, 0., 0.9, 1);
     c_jcircl(x-width/2, y-15.0, 0., 0.9, 1);
}

lbus(color, x, y, h)
int color;
float x, y, h;
{
int i; float t;
static float  width=.24, tx[3]={.45, .45, -.9}, ty[3]={-.5, .5, .0};
     c_jpintr(1);
   c_jpidex(color, 9);
   c_jcolor(color);
   c_jrect(x-width, y, x+64., y+width);
     for (i=0; i<5; i++) {
          t = i*16.+x;
          c_jrect(t-width, y-8., t, y);
      c_jrect(t-1.14, y-8.1, t+0.9, y-8.);
      c_jrect(t-1.14, y-8.5, t+0.9, y-8.6);
          c_jrect(t, y-h, t-width, y-8.6);
          c_jrmove(-.60, .0);
          c_jrplgn(tx, ty, 3);
     }
}

back(x, y)
float x, y;
{
   c_jpidex(CYAN , 900);
   c_jrect(87., 6., 98., 19.);
     c_jsize(0.8, 1.6);
     c_jrmove(.7, 2.);
     c_j1strg(" Breadboard");
     c_jrmove(0., 3.0);
     c_j1strg("     to ");
     c_jrmove(0., 3.0);
     c_j1strg("    Back");
}
/******** end of file "isub_pm.c"   *********/
```

Appendix D.1 Main program to present the graphics displays

```
/************* filename: laseps.c
                  by Kwangsoo Yi  **************/

#include "const.h"
#include "stdio.h"

main ()
{
float ratio, list[4];
int flag5=0, button, segnam, pickid, dumm1[5], dumm2[5];
     c_jbegin();
          c_jdinit(DSP);
          c_jdevon(DSP);
          c_jbgbat(3);
          c_jvport(-1.,1.,-.8,.8);
          c_jwindo(0.,100.,0.,80.);
    c_jfsopn(2, DSP, 9, "dbg");
          c_jsetdb(6);
             c_jdevwn(DSP,-1.,1.,-.8,.8);
             c_jescap(4111, 0, 0, dumm1, dumm2);/*dialog invisible*/
             c_jddete(1);
             c_jsrest(2);
               c_jvsall(0);
               c_jvisbl(1,1);
             c_jienab(DSP, 5, MOUSE);
               c_jenbat();
      for(;;){
             c_jpick(DSP, MOUSE, 1, &button, &segnam, &pickid);
             switch(pickid){
                  case 2:
                        sas();            /*  2.0  */
                       break;
                  case 3:
                        sas();            /*  3.0  */
                       break;
                  case 4:
                        bat();            /*  4.0  */
                       break;
                  case 5:
                        bat();            /*  5.0  */
                       break;
                  case 6:
                        amps_lc();        /*  6.0  */
                       break;
                  case 7:
                        p3();             /*  7.0  */
                       break;
                  case 8:
                        p3();             /*  8.0  */
                       break;
```

```
                case 9:
                        pmad();          /*  9.0  */
                    break;
                case 10:
              sumdata();        /*  10.0 */
                    break;
                case 11:                  /*  end  */
                        c_jescap(4110,   0,   0,   dumm1,   dumm2);/*dialog
visible*/
                        c_jidisa(DSP, 5, MOUSE);
                        c_jdevof(DSP);
                        c_jdend(DSP);
                        c_jend();
                        exit(0);
                    break;
          }
                c_jbgbat(3);
                   c_jvsall(0);
                 c_jvisbl(1,1);
          c_jenbat();
        }
}

/*******      end of file "laseps.c"    ********/
```

Appendix D.2 -- 2.0 SAS(1) and 3.0 SAS(2)

```
/************* filename: sas.c  ---  2.0     **************/

#include "math.h"
#include "const.h"
#include "stdio.h"

sas ()
{
static float rad, v, am, t1, t2;
int i, sunst, j, button, segnam, pickid;
char  str[20], suns[10];
FILE *fp;
     c_jbgbat(3);                    /* 2.1 Envisible segment 2  */
        c_jvisbl(1,0);
        c_jvisbl(2,1);
     c_jenbat();
     fp=fopen("sas.dat","r");        /* 2.2 Get input  */
        fscanf(fp, "%d,%f,%f,%f,%f ",&sunst, &am, &rad, &t1, &t2);

   c_jopen();                       /* 2.3 Display the chart */
        c_jsize(1.2, 1.9);
          orbit(rad, 16.,50.);
      c_jcolor(BLACK);
          c_jmove(78., 62.); c_jlstrg(sunst==0 ? " ECLIPSE": " SUN");
        c_jsize(1.0, 1.9);
          c_jrmove(0., -6.);
          c_jrmove(0., -6.);
          c_jrmove(0., -6.); c_jlstrg(strcat(f2a(am, 7, 2, str)," Amp"));
      c_jclose();
     c_jienab(DSP, 5, MOUSE);
     do{
        c_jpick(DSP, MOUSE, 1, &button, &segnam, &pickid);
        switch(pickid){
             case 1: return(1); break;
         }

      }while(pickid<8 && 0<pickid);
}

orbit(rad, x, y )
float rad, x, y ;
{
static float radius=14., x1[4]= {2.0, .0, -2.0, .0},
                                        y1[4]= {.3,  .3, -.3, -.3},
      x2[8]= {0.5, 1.0, .5, .0, -.5, -1.0, -.5, .0},
      y2[8]= {-.2,  .4, .2, .2,  .2, -.4, -.2, -.2};
     c_jmove(x+(radius+1.9)*cos(rad), y+(radius+1.9)*sin(rad));
     c_jrmove(0., -1.2);
     c_jrdraw(.0, 2.);
  c_jpintr(1); c_jpidex(GREEN,100);
```

```
      c_jrmove(-1.0, -.25);
      c_jrplgn(x1, y1, 4);
      c_jrmove(.0, -.6);
      c_jrplgn(x1, y1, 4);
      c_jrmove(.0, -.6);
    c_jpidex(WGREY,100);
      c_jrplgn(x2, y2, 8);
      c_jrmove(.0, -.6);
    c_jpidex(GREEN,100);
      c_jrplgn(x1, y1, 4);
}

/*******      end of file "sas.c"    ********/
```

Appendix D.3 -- 4.0 BAT(1) and 5.0 BAT(2)

```
/************* filename: bat1.c -  4.0 BAT  - **************/


#include "const.h"

bat()
{
int  i, j, nm, dum1[5], dum2[5];
static float volt[15][12];
     c_jbgbat(0);              /* 4.1 Envisible dialog area */
   c_jvsall(0);
     c_jescap(4115, 0, 0, dum1, dum2);/*dialog clear*/
     c_jescap(4110, 0, 0, dum1, dum2);/*dialog visib*/
     c_jenbat();
                              /* 4.3 Display data  */
          printf("\n**********  BATTERY  CELL  DATA  (Volts)
************************");
          for (i=1; i<15; i++){
         printf("\n%2d", i);
               for (j=0; j<12; j++){
                    volt[i][j]=1.+i*.01+.01*j;
                    printf("%6.3f", volt[i][j]);
                    }
               }
          printf("\n\n****  BATTERY  CELL  REFERENCE  DATA
************************\n");

   c_jpause(1);              /* 4.4 Pause */
     c_jescap(4111, 0, 0, dum1, dum2);/*dialog invisib*/
}

/*******     end of file "bat1.c"   ********/
```

Appendix D.4 -- 6.0 AMPS_LC

```
/************* filename: amps_lc.c --- 6.0   **************/

#include "const.h"
#include "stdio.h"

amps_lc ()
{
static float v, am, st, t;
static float volt[10][2], amp[10][2], swtemp[10][2], temp[10];
static int onof, onoff[10][2], on10_3, flag61=0;
int i, j, button, segnam, pickid;
char  str[20];
FILE *fp;
     c_jbgbat(3);                          /* 6.1 envisible segment 6 */
          c_jvsall(0);
          c_jvisbl(6, 1);
       if (flag61==1)  c_jvisbl(61, 1);
      c_jenbat();
     fp=fopen("1c.dat","r");               /* 6.2 get input  */
     for (i=0; i<10; i++){
        for (j=0; j<2; j++){
             fscanf(fp, "%d,%f,%f,%f,%f ",&onof,&v,&am,&st,&t);
             onoff[i][j]=onof;
           volt[i][j]=v;
             amp[i][j]=am;
             swtemp[i][j]=st;
         }
          temp[i]=t;
      }
     on10_3=0;
     if (flag61==0) {                     /* 6.3 Display the chart and diagram */
      c_jropen(61);
           c_jcolor(BLACK);
         c_jmove(9.8, 47.);
           for (i=0; i<9; i++){
             c_jrdraw(onoff[i][0]-1., 4.);
             c_jrmove(4.-onoff[i][0], -4.0);
             c_jrdraw(onoff[i][1]-1., 4.);
            c_jrmove(6.5-onoff[i][1], -4.0);
            }
           c_jrdraw(onoff[9][0]-1., 4.);
             c_jrmove(4.-onoff[9][0], -4.0);
             c_jrdraw(onoff[9][1]-1., 4.);
             c_jrmove(4.-onoff[9][1], -4.0);
             c_jrdraw(on10_3-1., 4.);

           c_jsize(0.8, 1.);
         c_jmove(12.0, 32.4);
           for (i=0; i<10; i++){
              for (j=0; j<2; j++){
```

```
                    if (onoff[i][j]==1) {
                  c_j1strg(f2a(volt[i][j], 7, 2, str));
                  c_jrmove(.0, -3.); c_j1strg(f2a(amp[i][j], 7, 2, str));
                  c_jrmove(2.0, -3.); c_j1strg(i2a(j+1,
str));c_jrmove(-2.0,.0);
                              }
                             else c_jrmove(0., -6.);
              c_jrmove(.0, -3.*(j+1)); c_j1strg(f2a(swtemp[i][j], 7, 2,
str));
              c_jrmove(.0, 3.*(3+j));
             }
              c_jrmove(.0, -3.*(3+j)); c_j1strg(f2a(temp[i], 7, 2, str));
              c_jrmove(8.5, 15.0);
          }
           flag61=1;
       c_jrclos();
        }
    c_jienab(DSP, 5, MOUSE);
    do{                                    /* 6.4 Input pickid  */
       c_jpick(DSP, MOUSE, 1, &button, &segnam, &pickid);
       switch(pickid){
            case 1: ldat(1);                /*  6.5 Data    */
                    c_jbgbat(3);  c_jvisbl(6,1);  c_jvisbl(61,1);
c_jenbat();
                            break;
            case 2: return(1); break;
        }
     }while(1);
}

/*******     end of file "amps_lc.c"   ********/
```

Appendix D.5 -- 7.0 P3.1 amd 8.0 P3.2

```
/************** filename: p3.c       **************/


p3()
{
int  segnam, pickid, dumm1[5], dumm2[5];
     c_jbgbat(0);
   c_jvsall(0);
     c_jescap(4115, 0, 0, dumm1, dumm2);/*dialog clear*/
     c_jescap(4110, 0, 0, dumm1, dumm2);/*dialog visib*/
     c_jenbat();
   c_jpause(1);
     c_jescap(4111, 0, 0, dumm1, dumm2);/*dialog invisib*/

}

/*******      end of file "p3.c"    ********/
```

Appendix D.6 -- 9.0 PMAD

```
/************* filename: pmad.c  --- 9.0     **************/

#include "const.h"
#include "stdio.h"

pmad ()
{
static float v, am, volt[2][2], amp[2][2];
int i, j, button, segnam, pickid;
char  str[20];
FILE *fp;
     c_jbgbat(3);                    /* 9.1 Envisble segment 9 */
          c_jvsall(0);
          c_jvisbl(9, 1);
      c_jenbat();
     fp=fopen("pmad.dat","r");      /* 9.2 Get input */
     for (i=0; i<2; i++){
        for (j=0; j<2; j++){
           fscanf(fp, "%f,%f ",&v,&am);
             volt[i][j]=v; amp[i][j]=am;
         }
      }
   for (;;) {
   c_jopen();                       /* 9.3 Display data */
        c_jcolor(BLACK); c_jsize(.8, 1.0);
          c_jmove(3.,  72.);  c_j1strg(strcat(f2a(volt[0][0],  7,  2,  str),"
V"));
          c_jrmove(0.,  -1.5);  c_j1strg(strcat(f2a(amp[0][0],  7,  2,  str),"
Amp"));
          c_jrmove(84.,  1.5);  c_j1strg(strcat(f2a(volt[1][0],  7,  2,  str),"
V"));
          c_jrmove(0.,  -1.5);  c_j1strg(strcat(f2a(amp[1][0],  7,  2,  str),"
Amp"));
          c_jmove(14.5,  62.5);  c_j1strg(strcat(f2a(volt[0][1],  7,  2,  str),"
V"));
          c_jrmove(0.,  -1.3);  c_j1strg(strcat(f2a(amp[0][1],  7,  2,  str),"
Amp"));
          c_jmove(86.5,  62.5);  c_j1strg(strcat(f2a(volt[1][1],  7,  2,  str),"
V"));
          c_jrmove(0.,  -1.3);  c_j1strg(strcat(f2a(amp[1][1],  7,  2,  str),"
Amp"));
      c_jclose();

     c_jienab(DSP, 5, MOUSE);
        c_jpick(DSP, MOUSE, 1, &button, &segnam, &pickid);
        switch(pickid){
             case 1:
             case 2:
             case 3:
             case 4:
```

```
            case 5:
                  pldcen(pickid);
             c_jbgbat(3); c_jvsall(0); c_jvisbl(9, 1); c_jenbat();
                 break;
            case 9: return(1); break;
         }
      }
}

pldcen(pid)                          /* --- 9.4 PLDCEN --- */
int pid;
{
static float v, am, volt[2][2], amp[2][2], current[2][5];
int i, j, button, segnam, pickid;
char  str[20];
FILE *fp;
     c_jbgbat(3);                    /*  9.4.1 Envisible segment 10 */
         c_jvsall(0);
         c_jvisbl(10, 1);
      c_jenbat();
     fp=fopen("subpm.dat","r");      /*  9.4.2 Get Input  */
     for (i=0; i<2; i++){
        for (j=0; j<2; j++){
           fscanf(fp, "%f,%f ",&v,&am);
             volt[i][j]=v; amp[i][j]=am;
               }
      }
     for (i=0; i<2; i++){
        for (j=0; j<5; j++){
           fscanf(fp, "%f ",&am);
             current[i][j]=am;
               }
      }
   c_jopen();                        /* 9.4.3 Display data */
        c_jcolor(BLACK); c_jsize(.7, 1.3);
      /*Bus A Data display*/
         c_jmove(15.,  68.);  c_jlstrg(strcat(f2a(volt[0][0],  7,  2,  str),"
V"));
         c_jrmove(0.,  -2.);  c_jlstrg(strcat(f2a(amp[0][0],  7,  2,  str),"
Amp"));
         c_jrmove(3., -11.);  c_jlstrg(strcat(f2a(volt[1][0],  7,  2,  str),"
V"));
         c_jrmove(0.,  -2.);  c_jlstrg(strcat(f2a(amp[1][0],  7,  2,  str),"
Amp"));
      c_jmove(12., 41.5);
        for (j=0; j<5; j++){
            c_jlstrg(strcat(f2a(current[0][j], 7, 2, str),"Amp"));
         c_jrmove(16., 0.);
               }
      /*Bus B Data display*/
         c_jmove(73.,  68.);  c_jlstrg(strcat(f2a(volt[0][1],  7,  2,  str),"
V"));
```

```
            c_jrmove(0.,  -2.);  c_j1strg(strcat(f2a(amp[0][1],  7,  2,  str),"
Amp"));
            c_jrmove(-3., -11.); c_j1strg(strcat(f2a(volt[1][1], 7, 2, str),"
V"));
            c_jrmove(0.,  -2.);  c_j1strg(strcat(f2a(amp[1][1],  7,  2,  str),"
Amp"));
        c_jmove(20., 22.5);
          for (j=0; j<5; j++){
               c_j1strg(strcat(f2a(current[1][j], 7, 2, str),"Amp"));
           c_jrmove(16., 0.);
                  }
        c_jclose();
    for (;;) {
          c_jpick(DSP, MOUSE, 1, &button, &segnam, &pickid);
          switch(pickid){
               case 1: return(1); break;
           }
        }
}
/*******      end of file "pmad.c"    ********/
```

Appendix D.7 -- 10.0 SUMMARY_DATA

```
/************* filename: sumdata.c  -- 10.0 Summary data **************/

sumdata()
{
int  segnam, pickid, dumm1[5], dumm2[5];
float svolt, samp, bamp, lamp, ch1a, ch2a, ch3a;
     c_jbgbat(0);
      c_jvsall(0);                                  /* 10.1  */
        c_jescap(4115, 0, 0, dumm1, dumm2);/*dialog clear*/
        c_jescap(4110, 0, 0, dumm1, dumm2);/*dialog visib*/
     c_jenbat();
          printf("********  POWER  SOURCE  CONTROLLER  SYSTEM  DATA
************\n");
          printf("\nSYSTEM VOLTAGE (Volts):");
          printf("%10.3f\n", svolt);
          printf("SOLAR ARRAY CURRENT (Amps):");
          printf("%10.3f\n", samp);
          printf("BATTREY CURRENT (Amps):");
          printf("%10.3f\n", bamp);
          printf("LOAD CURRENT (Amps):");
          printf("%10.3f\n", lamp);
          printf("CHANNEL 1 CURRENT (Amps):");
          printf("%10.3f\n", ch1a);
          printf("CHANNEL 2 CURRENT (Amps):");
          printf("%10.3f\n", ch2a);
          printf("CHANNEL 3 CURRENT (Amps):");
          printf("%10.3f\n", ch3a);
          printf("\n*****  BATTERY  TEMPERATURE  SENSOR  DATA  (DEG.  F)
*********\n");
          printf("\n");

  c_jpause(1);
     c_jescap(4111, 0, 0, dumm1, dumm2);/*dialog invisib*/
}

/*******     end of file "sumdata.c"   ********/
```

Appendix D.8 -- Load Center Data

```
/************* filename: ldat.c  - LOAD_CENTER/DATA**************/

ldat()
{
int  i, sw[10], ch[10], dumm1[5], dumm2[5];
float amp[10];
     c_jbgbat(3);
      c_jvsall(0);
        c_jescap(4115, 0, 0, dumm1, dumm2);/*dialog clear*/
        c_jescap(4110, 0, 0, dumm1, dumm2);/*dialog visib*/
     c_jenbat();
          printf("              SWITCH          CHANNEL          LOAD\n");
          printf(" RPC            ON            CONNECTION     CURRENT\n");

printf("-------------------------------------------------------\n");
          for (i=0; i<10; i++) {
               printf(" %2d   %9d      %9d   %7.2f\n", i+1, sw[i], ch[i],
amp[i]);
               }
          printf("\n");
          printf("\n  CHANNEL          DESIRED                RESULTING
RESULTING\n");
          printf(" NUMBER      LOADING          LOADING      MISMATCH\n");

printf("-------------------------------------------------------\n");
   c_jpause(1);
     c_jescap(4111, 0, 0, dumm1, dumm2);/*dialog invisib*/
}

/*******     end of file "ldat.c"   ********/
```

Appendix D.9 -- Library functions

```
/************* filename: lib1.c    **************/

#include "math.h"

char *i2a(n, str)
int n;  char str[];
{int q, i, rem;
   i = (n<10)? 1: log10((float)n)+1;
      str[i] = '\0';
        do {
         q = n/10;
           rem = n - q*10;
           n = q;
           str[--i] = rem + '0';
      } while(i>0);
   return(str);
}

char *f2a(r, l, d, str)
float r; register int l, d;  char str[];
{register int n, q, rem; int j;
     strcpy(str,"          ");
   for (j=0; j<d; j++) r *= 10;
   n = r;
   str[l] = '\0';
   for (j=0; j<d; j++) {
      q = n/10;
      rem = n - q*10;
      n = q;
      str[--l] = rem + '0';
      }
   str[--l] = '.';
   do {
      q = n/10;
      rem = n - q*10;
      n = q;
      str[--l] = rem + '0';
      } while(n>0);
   return(str);
}

/*******     end of file "lib1.c"   ********/
```

Appendix D.10 -- Include file

```
/************* filename: const.h   **************/

#define DSP    1
#define RED    1
#define GREEN  2
#define YELLO  3
#define BLUE   4
#define MAGEN  5
#define CYAN   6
#define WHITE  7
#define BLACK  8
#define WGREY   15
#define GREY   11
#define ORANG  60
#define MOUSE  3
#define ON     1
#define OFF    0
/********     end of file "const.h"   ********/
```