09:00:55 OCA PAD INITIATION - PROJECT HEADER INFORMATION 10/02/89

Active

Project #: A-8474 Cost share #: Rev #: 0
Center # : 24-5-R8474-000 Center shr #: OCA file #:
Work type : RES
Contract#: AGMT. DTD. 9/26/89 Mod #: Document : AGR
Prime #: Contract entity: GTRC

Subprojects ? : N
Main project #:

Project unit: EMSL-MSD Unit code: 01.021.212
Project director(s):
GOOCH J W EMSL-MSD (404)894-8485

Sponsor/division names: DR A C BROWN / ATLANTA, GA
Sponsor/division codes: 251 / 022

Award period: 891001 to 900131 (performance) 900430 (reports)

Sponsor amount	New this change	Total to date
Contract value	17,893.00	17,893.00
Funded	17,893.00	17,893.00
Cost sharing amount		0.00

Does subcontracting plan apply ?: N

Title: PRESSURE SENSOR FOR COMPUTER-ASSISTED ORTHOPEDIC DIAGNOSIS AND TREATMENT

PROJECT ADMINISTRATION DATA

OCA contact: David B. Bridges 894-4820

Sponsor technical contact

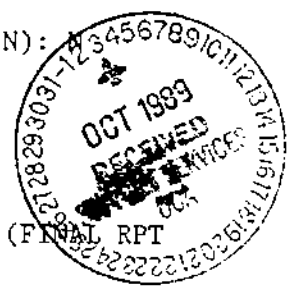
ALGIE BROWN, M.D.
(404)256-2511
960 JOHNSON FERRY RD.
SUITE 534
ATLANTA, GA 30342

Sponsor issuing office

SAME
(000)-

Security class (U,C,S,TS) : U
Defense priority rating : N/A
Equipment title vests with: Sponsor

ONR resident rep. is ACO (Y/N): N
supplemental sheet
GIT X



Administrative comments -
PROJECT INITIATION, FUNDS IAO $17,893, TERMINATION DATE OF 1/31/90 (FINAL RPT DUE 4/30/90), ADV. PAYMT OF $4500 RECEIVED

GEORGIA INSTITUTE OF TECHNOLOGY
OFFICE OF CONTRACT ADMINISTRATION

NOTICE OF PROJECT CLOSEOUT

Closeout Notice Date 06/29/90

Project No. A-8474 Center No. 24-6-R8474-000

Project Director GOOCH J W School/Lab EMSL-MSD

Sponsor DR A C BROWN/ATLANTA, GA

Contract/Grant No. AGMT. DTD. 9/26/89 Contract Entity GTRC

Prime Contract No.

Title PRESSURE SENSOR FOR COMPUTER-ASSISTED ORTHOPEDIC DIAGNOSIS AND TREATMENT

Effective Completion Date 900531 (Performance) 900831 (Reports)

Closeout Actions Required:	Y/N	Date Submitted
Final Invoice or Copy of Final Invoice	Y	900425
Final Report of Inventions and/or Subcontracts	Y	
Government Property Inventory & Related Certificate	N	
Classified Material Certificate	N	
Release and Assignment	N	
Other	N	

Comments

Subproject Under Main Project No.

Continues Project No.

Distribution Required:

Project Director	Y
Administrative Network Representative	Y
GTRI Accounting/Grants and Contracts	Y
Procurement/Supply Services	Y
Research Property Managment	Y
Research Security Services	N
Reports Coordinator (OCA)	Y
GTRC	Y
Project File	Y
Other	N
	N

NOTE: Final Patent Questionnaire sent to PDPI.

FINAL REPORT

PRESSURE SENSOR FOR COMPUTER ASSISTED ORTHOPEDIC DIAGNOSIS AND TREATMENT

GTRI PROJECT A8474

Prepared for

A. C. Brown, M.D.
960 Johnson Ferry Road, Suite 534
Atlanta, Georgia 30342

Contact: A. C. Brown, M.D.
Telephone: (404)256-2511
FAX: (404)256-0976

Prepared by

Georgia Tech Research Institute
Energy and Materials Sciences Laboratory
Atlanta, Georgia 30332

Contacts: Jan W. Gooch
James W. Larsen
Telephone: (404)894-8485
FAX: (404)894-6199

GEORGIA INSTITUTE OF TECHNOLOGY
A Unit of the University System of Georgia
Atlanta, Georgia 30332

GTRI
GEORGIA TECH RESEARCH INSTITUTE

FINAL REPORT

PRESSURE SENSOR FOR COMPUTER ASSISTED ORTHOPEDIC DIAGNOSIS AND TREATMENT

GTRI PROJECT A8474

Prepared for

A. C. Brown, M.D.
960 Johnson Ferry Road, Suite 534
Atlanta, Georgia 30342

Contact: A. C. Brown, M.D.
Telephone: (404)256-2511
FAX: (404)256-0976

Prepared by

Georgia Tech Research Institute
Energy and Materials Sciences Laboratory
Atlanta, Georgia 30332

Contacts: Jan W. Gooch
James W. Larsen
Telephone: (404)894-8485
FAX: (404)894-6199

June 11, 1990

TABLE OF CONTENTS

1.0 INTRODUCTION

1.1 Background

The diagnosis and treatment of physical impairment of the foot is presently a highly empirical practice, relying on the subjective judgement of the attending physician. Treatment frequently involves the prescription of various orthotic devices which redistribute or redirect the loads placed on the various parts of the patient's foot. In order to accurately and reliably measure the pressure distribution on a patient's foot, the Sponsor initiated a research project with the Georgia Tech Research Institute (GTRI) to produce a device which would measure this pressure distribution of feet. Also, the device would provide a visual display and computer-readable data file which would be stored and analyzed for the purposes of treatment and charting.

This technology was developed at the Georgia Tech Research Institute for applications in robotic tactile sensing.[1] A comprehensive review of tactile sensing has recently been published[2] and, therefore, only the aspects relevant to the present application will be considered here.

The function of any pressure sensor is to determine both the spatial location and magnitude of forces resulting from contact with an object. Currently, available sensors can typically determine only the spatial coordinates of an applied force, and are limited by their rigidity and lack of resolution.

2.0 GOALS

2.1 Development of Foot-Pressure Sensing System

The over-all system will sense pressure on both feet simultaneously, graphically display the pressure on a computer monitor, and store the data so that it can be accessed at any time in the future.

2.2 Deliverable Items

2.2.1 Prototype system: One prototype pressure sensing system incorporating two eight-inch by sixteen-inch sensing pads was delivered to the Sponsor at the conclusion of this program. The deliverable hardware included the portable sensor and data collection circuitry only. The commercially available computer interface board and the computer itself will be supplied by the Sponsor.

2.2.2 Formulation: The formulation of the curing conditions of the conductive polymeric section of the sensor pad has been delivered.

2.2.3 Schematic diagrams of circuits: The schematic diagrams of all circuits for data collection has been delivered.

2.2.4 Computer program: All computer programs for this project have been delivered.

2.2.5 Manuals, drawings and specifications: All manuals, drawings and specifications for reproducing and operating the system have been delivered.

2.2.6 Final report: A final report has been prepared and contains the above deliverable items with the exception of hardware items.

3.0 EXPERIMENTATION

The experimentation segment of this project consisted of five major parts as follows:

3.1 Circuit Fabrication

The over-all system is shown in Figure 1 - System Block Diagram. The diagram shows the connection of each board.

3.1.1 Buffer board - See Appendix A

3.1.2 Row driver board - See Appendix A

3.1.3 Column detection board - See Appendix A

3.1.4 Pressure sensing boards - See Appendix B

The pressure sensing boards required the design and fabrication of two circuit boards approximately eight inches wide (32 contacts) by sixteen inches long (64 contacts) with measuring electrodes spaced on 0.25 inch centers. These were suitable for a person of 20 to 350 pounds in weight in sitting or standing positions.

3.2 Elastomer Formulation

This task consisted of the determination of the proper elastomer formulation and curing conditions for sensing of pressures in the required pressure range. Pressure that was calculated to be encountered was in the range of approximately one to fifty pounds per square inch.

The conductive and pressure sensitive elastomer was formulated to possess a "reduction of d.c. resistance with pressure" relationship. The elastomer formulation was coated (1.0 mil thickness) on a plastic film substrate and cured. The formulation is contained in Table 1.

TABLE 1. Formulation of Conductive Elastomer

COMPONENT	MANUFACTURER	% WEIGHT
Poly BD (Butudiene Prepolymer)	ARCO Corporation	31.5
ISONATE 143L (Diisocyanate Prepolymer)	DOW Chemical Corporation	1.0
Carbon, electronic grade	Cabot Corporation	67.0

The above mixture was diluted with solvent to form a solution consisting of 30 percent toluene.

The formulation in Table 1 was liquid applied with a wire-coater onto the plastic Ultem® film. The coated film was over-layed on the pressure sensing boards to physically contact the conducting surfaces of each board. The surface resistivity of the cured coating was 5.0 ohms-cm at 0.0 psi.

3.3 Computer Program

A computer program was developed to sense the pressure from the device, display the image and store the data. The program was written in C - Machine Language and is contained in Appendix C.

4.0 RESULTS

The system measured the pressure of human feet using ten individuals ranging in body weight from 90 psi to 220 psi. The pressure of both feet were measured and displayed simultaneously on a color monitor driven by an AT IBM personal computer. The results of measurements and reproducibility are listed in Table 2.

5.0 CONCLUSIONS

The foot pressure sensing device is capable of measuring pressure distribution for one or both feet which were the primary objectives of the project. The 0.25 inch distance between sensing points appears to be adequate for profiling the pressure under the feet although the distance could be reduced to 0.12 inch for greater accuracy. We recommend that the device be used for actual patients and that GTRI be given feed-back for further improvement of the technology.

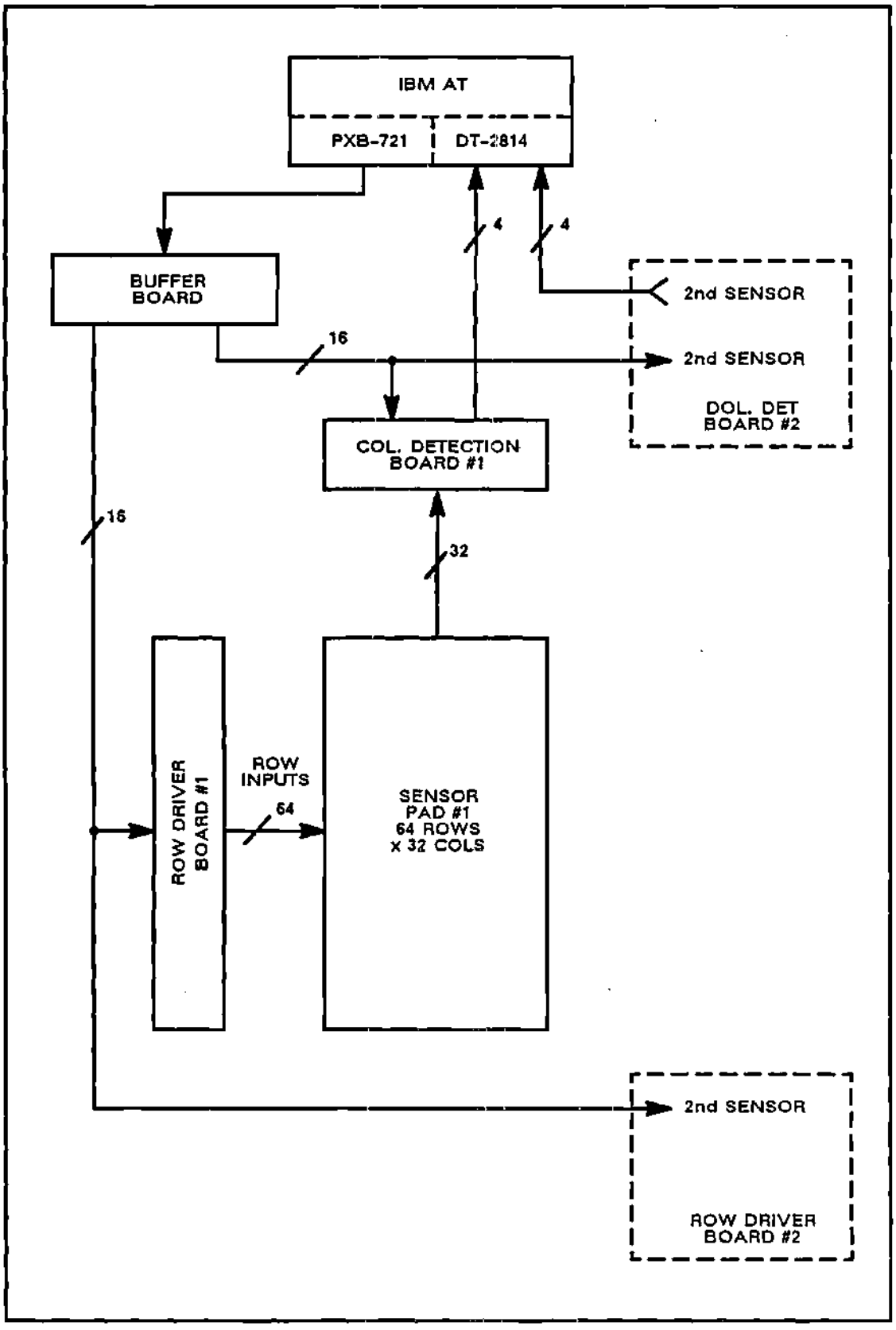
TABLE 2. Parameters, Measurements and Reproducibility

Parameter	Measurement	Reproducibility
Pressure Range per Pixel	0 - 50 psi	-
Load Range per Pixel	0 - 200 pounds	-
Foot Size (Toe to Heel)	2 in. - 14 in.	-
Total Body Weight	0 - 300 pounds	+/- 3 pounds

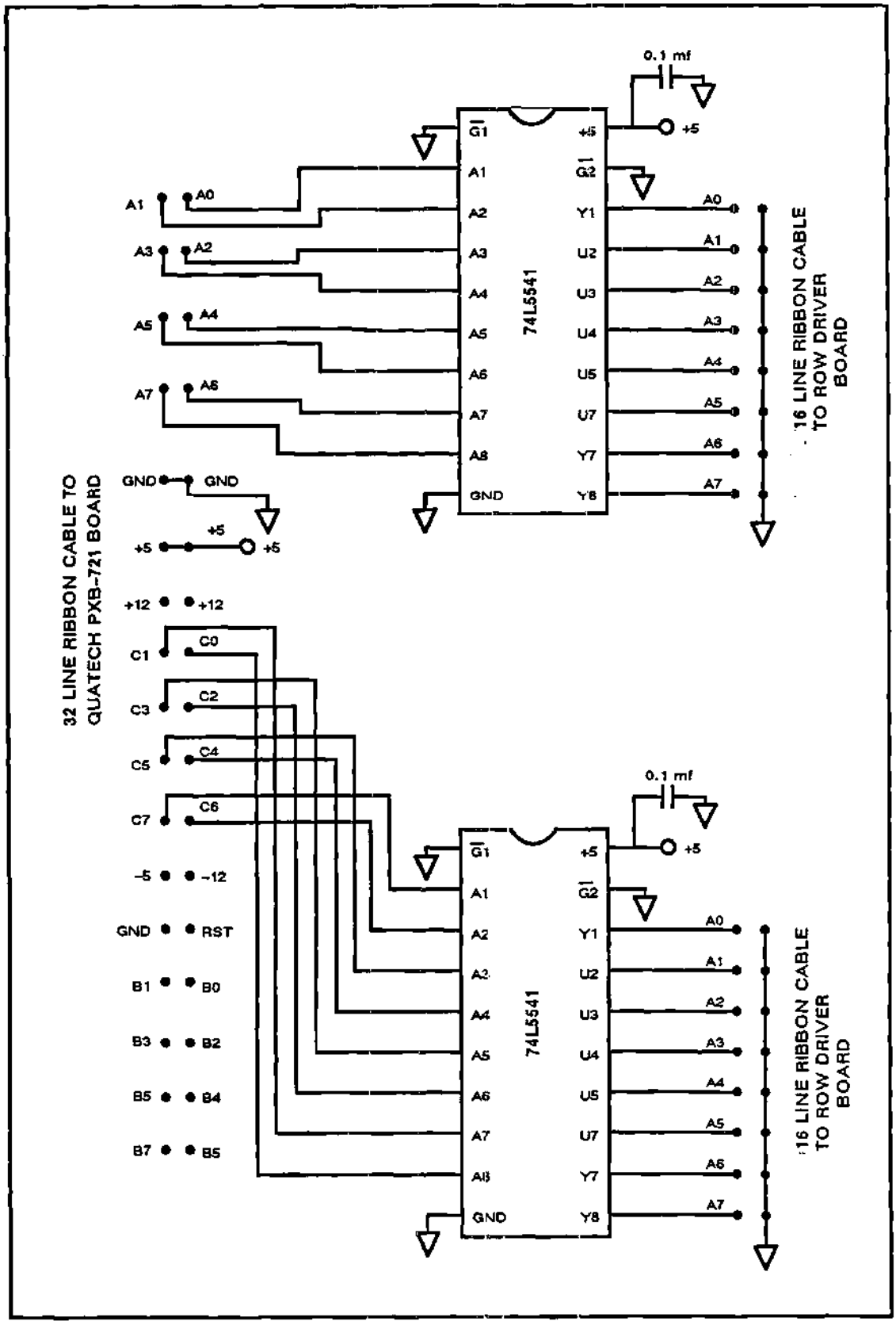
APPENDIX A

Schematic Diagrams of Circuit Boards

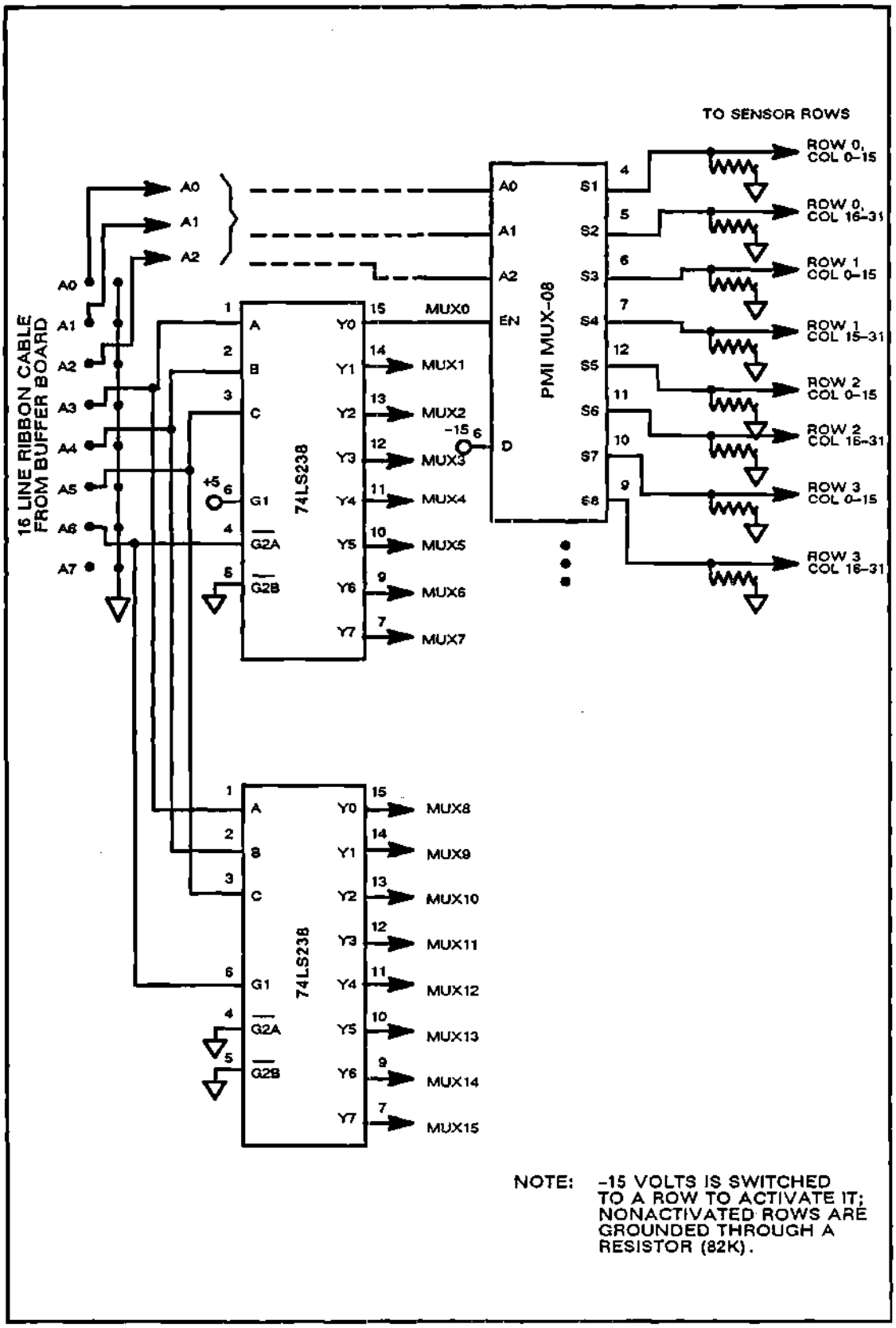
SYSTEM BLOCK DIAGRAM



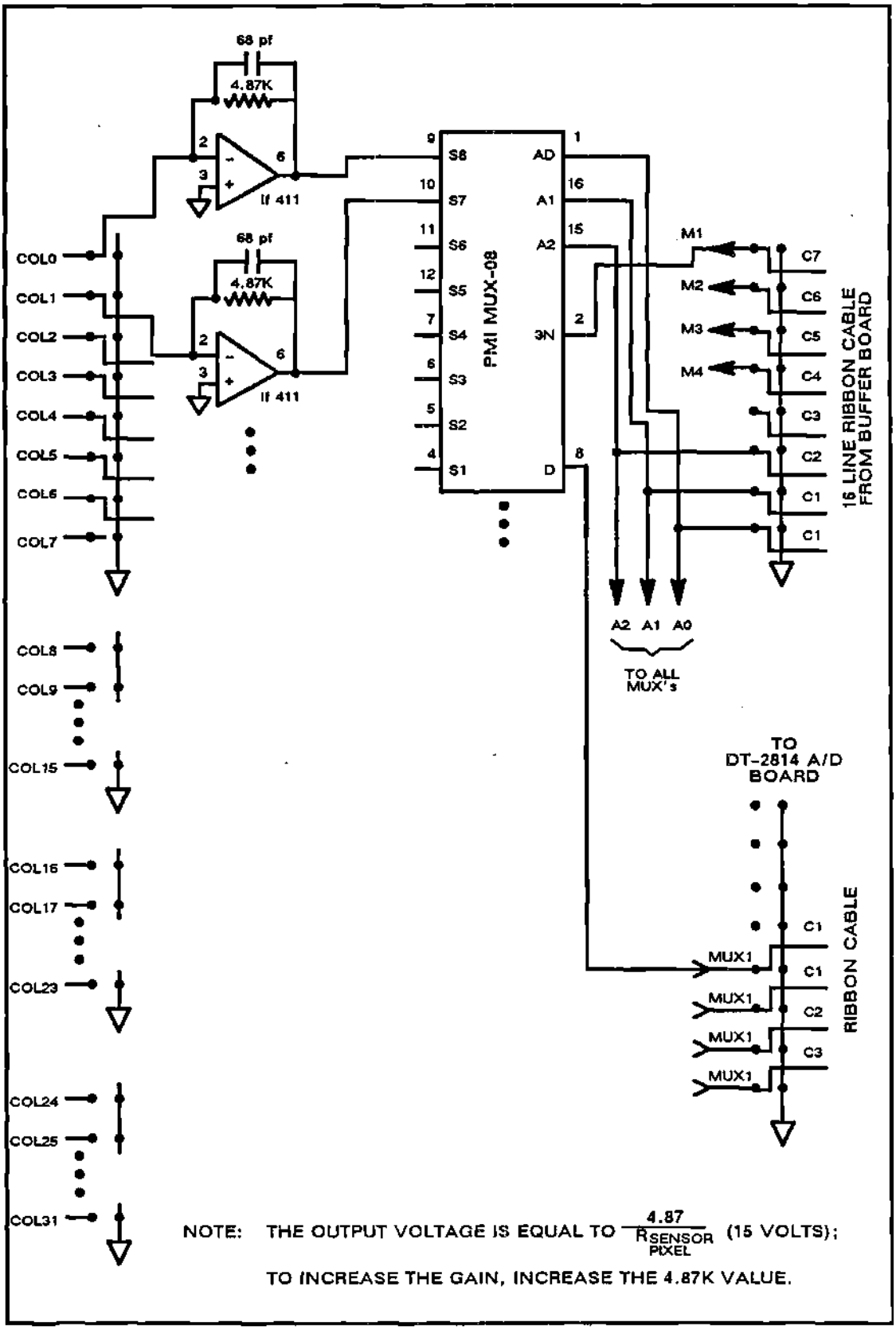
BUFFER BOARD



ROW DRIVER BOARD



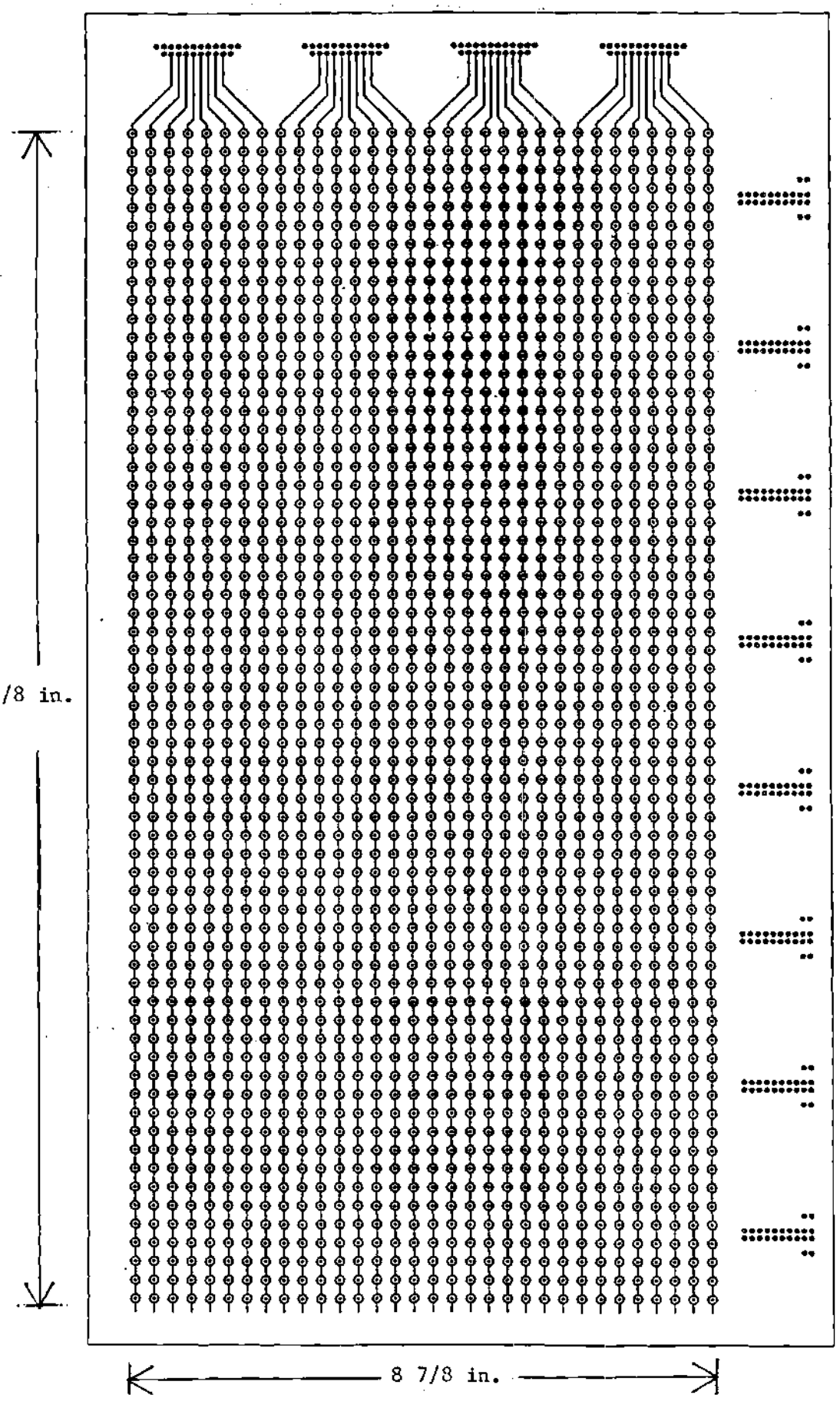
COLUMN DETECTION BOARD



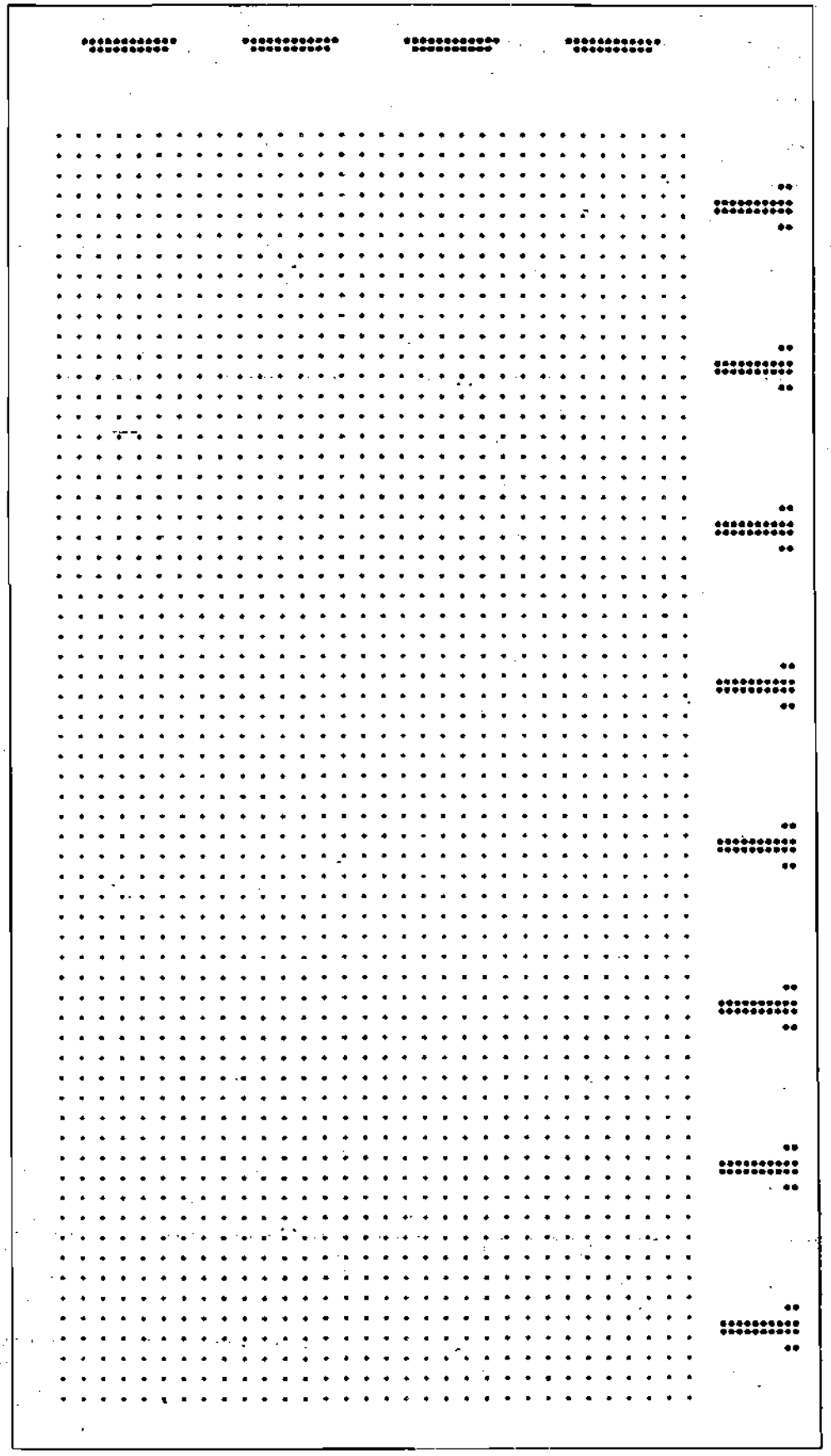
APPENDIX B

Schematic Diagrams of Pressure Sensing Boards

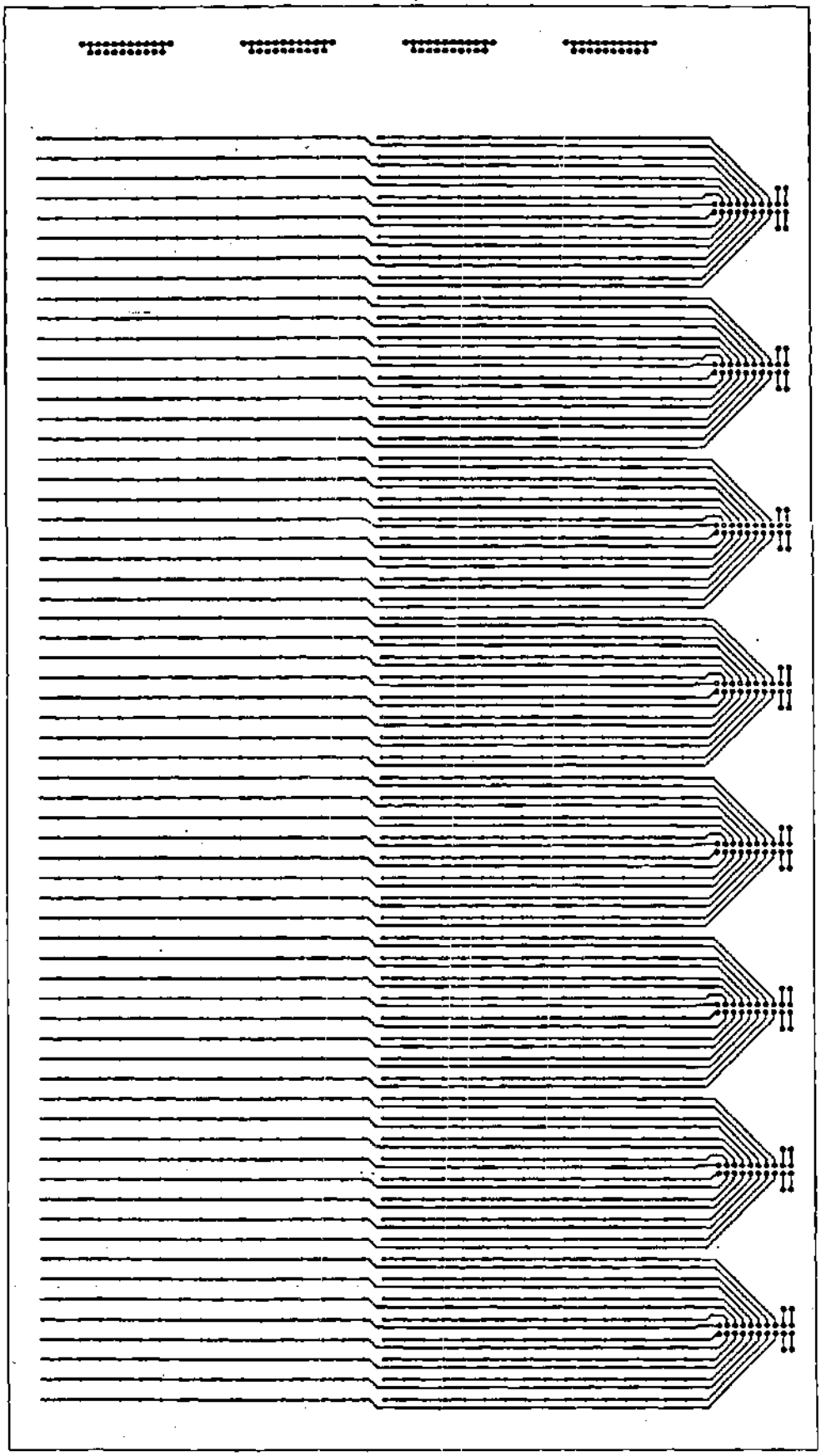
TOP SIDE OF SENSOR BOARD



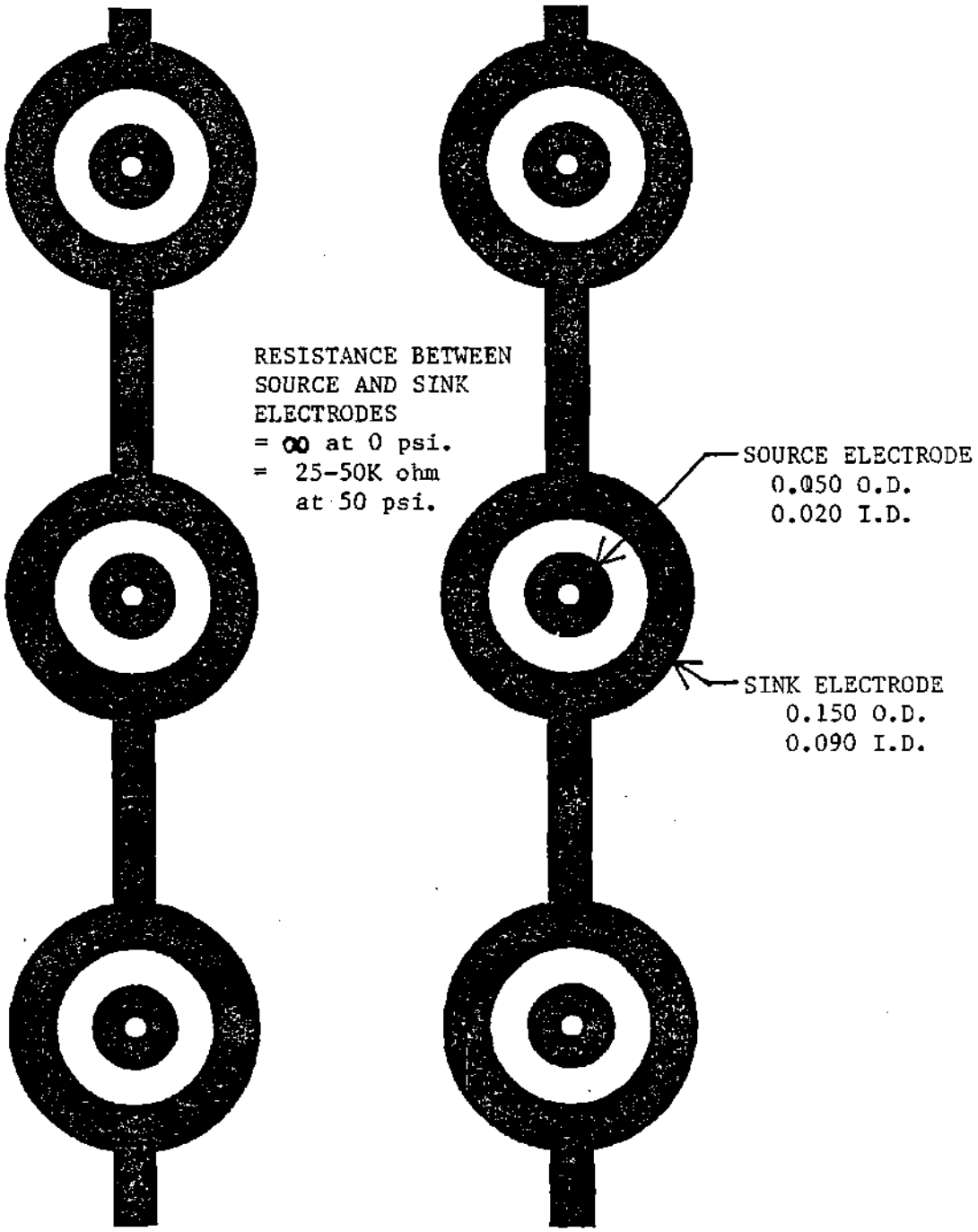
CENTER INTERCONNECTS



BOTTOM SIDE OF SENSOR BOARD



SOURCE AND SINK ELECTRODES



APPENDIX C

Computer Program

```
/* File foot4.c last update 04-05-1990 at 23:00 */

/* This program drives the quatech 72 line parallel board
   and the DT-2814 adc board to control the tactile foot pad*/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <alloc.h>
#include <graphics.h>
#include <conio.h>
#include <process.h>

int rstqua();
int rstdt();
int zerout();
int inpres();
int foot1(int x,int y,int pres);
int foot2(int x,int y,int pres);
int colorkey(int x,int y);

unsigned int b1bs,b2bs;
unsigned int b1c1,b1c2,b1c3,b2c1,b2c2,b2c3;
unsigned int byt0,byt1,byt2,byt3,byt4,byt5,byt6,byt7,byt8;
unsigned int adcr,adsr,addata;
unsigned int upres;
unsigned int rowmux,colmux,colad;
int ppres;
int rm;

unsigned int bytfo,bytc,bytdac;
unsigned char ctbt;
unsigned char tcby;
unsigned int nbyt;
unsigned int ncyc;
unsigned char f0hr,f0ha,f01r,f01a,f0f;
unsigned char f1hr,f1ha,f11r,f11a,f1f;
unsigned char f2hr,f2ha,f21r,f21a,f2f;
unsigned char f3hr,f3ha,f31r,f31a,f3f;
unsigned char dc1b,dc1a,dc2b,dc2a;
int nr1;
unsigned int nde1;


main()
  {
  char flnm[80],flnm1[80];
  FILE *flp;
  FILE *flp1;
  char comm1[80];
  int pix[64][4];
  int fib[4];
  int conv[64];
  int col[4];
```

```
  unsigned char drv[20400];
  unsigned char bm[8],ibm[8];
  unsigned char c0l,c0h,c1l,c1h,c2l,c2h,c3l,c3h;
  unsigned char cal,cah,cao;
  int dac1,dac2,dac,dcb1,dca1;
  int i,j,jj,k,l;
  int bsc,base;
  int bt,bitn,btp;
  int a1,a2,a3,a4,a5,a6,a7,a8,a9,a10;
  int aa[64][32];
  unsigned int rseg,ui;
  int ncyci;
  double temp;
  float freq;
  int gdrv,gmod;/*graphics control codes*/
  int pres;/*pressure indicator for plot color*/
  /************* end of variable list **************/


    /************* setup QuaTech addresses and control ***********/
  b1bs=0x0300;/*** board #1 quatech base address ***/
  byt0=b1bs+0;
  byt1=b1bs+1;
  byt2=b1bs+2;
  byt3=b1bs+4;
  byt4=b1bs+5;
  byt5=b1bs+6;
  byt6=b1bs+8;
  byt7=b1bs+9;
  byt8=b1bs+10;
  b1c1=b1bs+3;/*** port control registers ***/
  b1c2=b1bs+7;
  b1c3=b1bs+11;

  ctbt=0x80;/*** 80h sets all ports as outputs ***/

  b2bs=0x0320;/*** board #2 dt-2814 base address ***/
  adcr=b2bs+0;
  adsr=b2bs+0;
  addata=b2bs+1;


/******** power on warning **********/
  printf("************************************\n");
  printf("**** THE DEVICE POWER MUST BE ON ****\n");
  printf("***** TURN ON THE DEVICE POWER ******\n");
  printf("************************************\n");



  /************ initialize QuaTech board **************/
  rstqua();
  zerout();
```

```
  /************ initialize DT-2814 adc board **********/
  rstdt();


  /*********** Set up for file to write pixel force data ***********/
  printf("Enter output file <path:>file_name.ext :");
  scanf("%s",flnm);
  flp=fopen(flnm,"w+t");/*open a file for ascii writing*/

  nrl=0;
  while(nrl<=0)
     {
     printf("Enter # of cycles per section:");
     scanf("%d",&nrl);
     gets(comm1);
     printf("# of cycles=%d\n",nrl);
     }

  fprintf(flp,"%d\n",nrl);




     gets(comm1);
  gdrv=EGA;
  gmod=EGAHI;
  detectgraph(&gdrv,&gmod);
  printf("drv=%d   mode=%d\n",gdrv,gmod);
  initgraph(&gdrv,&gmod,"");
  setcolor(CYAN);
/*  line(10,10,100,10);
  line(20,20,20,300);
*/
  settextstyle(0,0,1);
  colorkey(80,30);

  settextstyle(0,0,2);
  setcolor(WHITE);
  outtextxy(280,20,"Foot Pressure");
  settextstyle(0,0,1);
  outtextxy(265,50,"Left Foot");
  outtextxy(435,50,"Right Foot");


  for(k=0;k<nrl;k++)
     {
     /*********** read foot 1 data **********/
     fprintf(flp,"foot #1\n");
     printf("doing foot #1\n");
     for(i=0;i<8;i++)
        {
        rm=14-(2*i);
        for(j=0;j<8;j++)
```

```
{
rowmux=(unsigned int)(rm*8+2*j);
colad=(unsigned int)(3*32+16+0);
inpres();
for(jj=0;jj<8;jj++)
   {
   colmux=(unsigned int)(128+(7-jj));
   inpres();
   ppres=(int)(((float)(upres))/16-40);
   aa[8*i+j][0+jj]=ppres;
   if(ppres<=0)
      {
      ppres=0;
      }
      else
      {
      }
   if(ppres==0)
      {
      pres=0;
      }
      else
      {
      ppres=ppres/32+1;
      if(ppres>15)
         {
         ppres=15;
         }
      pres=(int)(ppres);
      }
   foot1(jj,i*8+j,pres);
   }
colad=(unsigned int)(3*32+16+1);
inpres();
for(jj=0;jj<8;jj++)
   {
   colmux=(unsigned int)(64+(7-jj));
   inpres();
   ppres=(int)(((float)(upres))/16-40);
   aa[8*i+j][8+jj]=ppres;
   if(ppres<=0)
      {
      ppres=0;
      }
      else
      {
      }
   if(ppres==0)
      {
      pres=0;
      }
      else
      {
      ppres=ppres/32+1;
      if(ppres>15)
```

```
            {
            ppres=15;
            }
         pres=(int)(ppres);
         }
      foot1(8+jj,i*8+j,pres);
      }
   rowmux=rowmux+1;
   colad=(unsigned int)(3*32+16+2);
   inpres();
   for(jj=0;jj<8;jj++)
      {
      colmux=(unsigned int)(32+(7-jj));
      inpres();
      ppres=(int)(((float)(upres))/16-40);
      aa[8*i+j][16+jj]=ppres;
      if(ppres<=0)
         {
         ppres=0;
         }
         else
         {
         }
      if(ppres==0)
         {
         pres=0;
         }
         else
         {
         ppres=ppres/32+1;
         if(ppres>15)
            {
            ppres=15;
            }
         pres=(int)(ppres);
         }
      foot1(16+jj,i*8+j,pres);
      }
   colad=(unsigned int)(3*32+16+3);
   inpres();
   for(jj=0;jj<8;jj++)
      {
      colmux=(unsigned int)(16+(7-jj));
      inpres();
      ppres=(int)(((float)(upres))/16-40);
      aa[8*i+j][24+jj]=ppres;
      if(ppres<=0)
         {
         ppres=0;
         }
         else
         {
         }
      if(ppres==0)
         {
```

```
                pres=0;
                }
                else
                {
                ppres=ppres/32+1;
                if(ppres>15)
                   {
                   ppres=15;
                   }
                pres=(int)(ppres);
                }
             foot1(24+jj,i*8+j,pres);
             }
          }
       }

for(i=0;i<64;i++)
   {
   for(j=0;j<32;j++)
      {
      fprintf(flp,"%4d ",aa[i][j]);/*write out pixel pressure data*/
      }
   fprintf(flp,"\n");/*write out line feed*/
   }

   /*********** read foot 2 data **********/
   fprintf(flp,"foot #2\n");
   printf("doing foot #2\n");
   for(i=0;i<8;i++)
      {
      rm=14-(2*i);
      for(j=0;j<8;j++)
         {
         rowmux=(unsigned int)(rm*8+2*j);
         colad=(unsigned int)(3*32+16+4);
         inpres();
         for(jj=0;jj<8;jj++)
            {
            colmux=(unsigned int)(128+(7-jj));
            inpres();
            ppres=(int)(((float)(upres))/16-40);
            aa[8*i+j][0+jj]=ppres;
            if(ppres<=0)
               {
               ppres=0;
               }
               else
               {
               }
            if(ppres==0)
               {
               pres=0;
               }
               else
               {
```

```
        ppres=ppres/32+1;
        if(ppres>15)
            {
            ppres=15;
            }
        pres=(int)(ppres);
        }
    foot2(jj,i*8+j,pres);
    }
colad=(unsigned int)(3*32+16+5);
inpres();
for(jj=0;jj<8;jj++)
    {
    colmux=(unsigned int)(64+(7-jj));
    inpres();
    ppres=(int)(((float)(upres))/16-40);
    aa[8*i+j][8+jj]=ppres;
    if(ppres<=0)
        {
        ppres=0;
        }
        else
        {
        }
    if(ppres==0)
        {
        pres=0;
        }
        else
        {
        ppres=ppres/32+1;
        if(ppres>15)
            {
            ppres=15;
            }
        pres=(int)(ppres);
        }
    foot2(8+jj,i*8+j,pres);
    }
rowmux=rowmux+1;
colad=(unsigned int)(3*32+16+6);
inpres();
for(jj=0;jj<8;jj++)
    {
    colmux=(unsigned int)(32+(7-jj));
    inpres();
    ppres=(int)(((float)(upres))/16-40);
    aa[8*i+j][16+jj]=ppres;
    if(ppres<=0)
        {
        ppres=0;
        }
        else
        {
        }
```

```
            if(ppres==0)
               {
               pres=0;
               }
               else
               {
               ppres=ppres/32+1;
               if(ppres>15)
                  {
                  ppres=15;
                  }
               pres=(int)(ppres);
               }
            foot2(16+jj,i*8+j,pres);
            }
         colad=(unsigned int)(3*32+16+7);
         inpres();
         for(jj=0;jj<8;jj++)
            {
            colmux=(unsigned int)(16+(7-jj));
            inpres();
            ppres=(int)(((float)(upres))/16-40);
            aa[8*i+j][24+jj]=ppres;
            if(ppres<=0)
               {
               ppres=0;
               }
               else
               {
               }
            if(ppres==0)
               {
               pres=0;
               }
               else
               {
               ppres=ppres/32+1;
               if(ppres>15)
                  {
                  ppres=15;
                  }
               pres=(int)(ppres);
               }
            foot2(24+jj,i*8+j,pres);
            }
         }
      }

   }

for(i=0;i<64;i++)
   {
   for(j=0;j<32;j++)
      {
      fprintf(flp,"%4d ",aa[i][j]);/*write out pixel pressure data*/
```

```
        }
      fprintf(flp,"\n");/*write out line feed*/
      }


/*
  for(i=0;i<32;i++)
      {
      pres=i;
      for(j=0;j<64;j++)
         {
         pres=(2*(i-14)*(i-14)+(j-32)*(j-32))/10;
         while(pres>15)
            {
            pres=pres-16;
            }
         foot1(i,j,pres);
         }
      }

  for(i=0;i<32;i++)
      {
      pres=i;
      for(j=0;j<64;j++)
         {
         pres=(2*(i-14)*(i-14)+(j-32)*(j-32))/20;
         while(pres>15)
            {
            pres=pres-16;
            }
         foot2(i,j,pres);
         }
      }
*/
      gets(comm1);
restorecrtmode();

  fflush(flp);
  fclose(flp);

  exit(0);



  }


  /*******************************************************/
  /************** functions ******************************/
  /*******************************************************/

  int rstqua()
  /* This function sets up the quatech board with all ports as outputs *
     {
     asm    push    ax
```

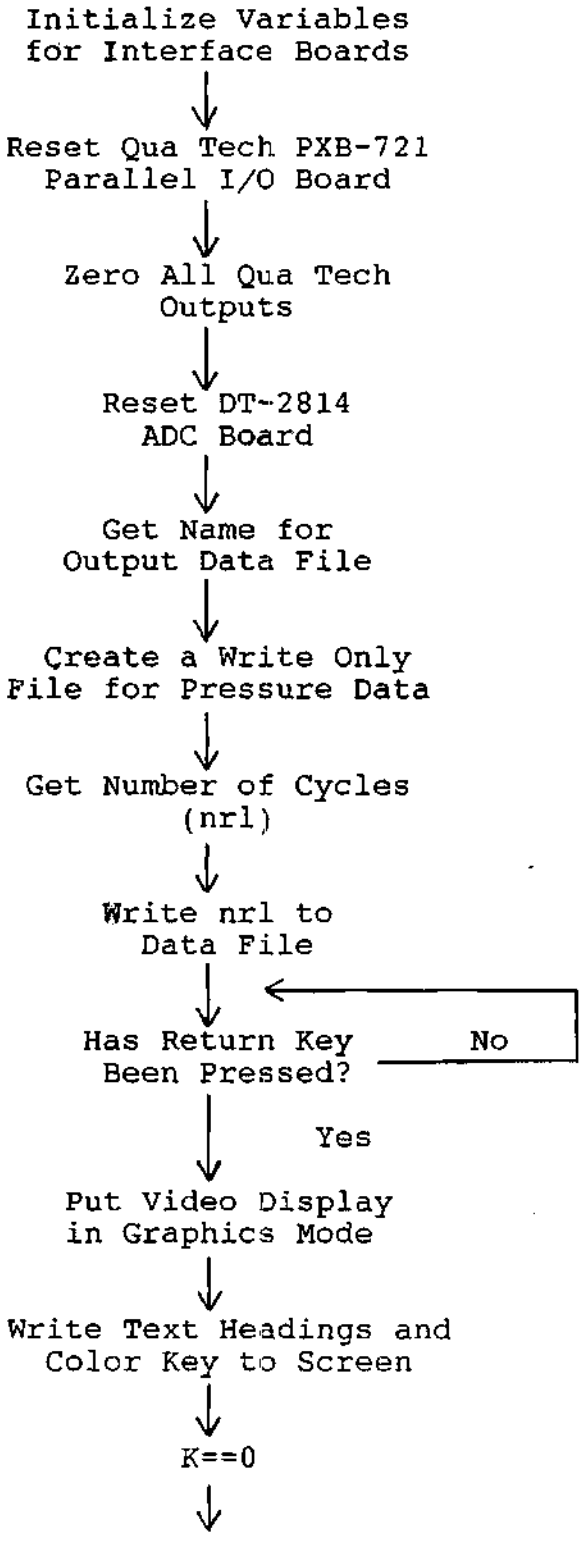
```
    asm     push    dx
    asm     mov     al,ctbt
    asm     mov     dx,b1c1
    asm     out     dx,al
    asm     mov     dx,b1c2
    asm     out     dx,al
    asm     mov     dx,b1c3
    asm     out     dx,al
    asm     pop     dx
    asm     pop     ax
    }

int rstdt()
/* This function resets the DT-2814 adc board */
    {
    asm     push    ax
    asm     push    dx
    asm     push    cx
    asm     mov     al,0
    asm     mov     dx,adcr
    asm     out     dx,al

    asm     mov     cx,100  /* wait 100 micro seconds */
    rstd1:
    asm     cmp     cx,0
    asm     dec     cx
    asm     jne     rstd1

    asm     mov     dx,addata
    asm     in      al,dx
    asm     mov     ah,al
    asm     in      al,dx   /* read in the high and low adc data bytes *
    asm     pop     cx
    asm     pop     dx
    asm     pop     ax
    }

int zerout()
/* This function zeros all of the outputs */
    {
    asm     push    ax
    asm     push    dx
    asm     mov     al,0
    asm     mov     dx,byt0
    asm     out     dx,al
    asm     mov     dx,byt1
    asm     out     dx,al
    asm     mov     dx,byt2
    asm     out     dx,al
    asm     mov     dx,byt3
    asm     out     dx,al
    asm     mov     dx,byt4
    asm     out     dx,al
    asm     mov     dx,byt5
    asm     out     dx,al
```

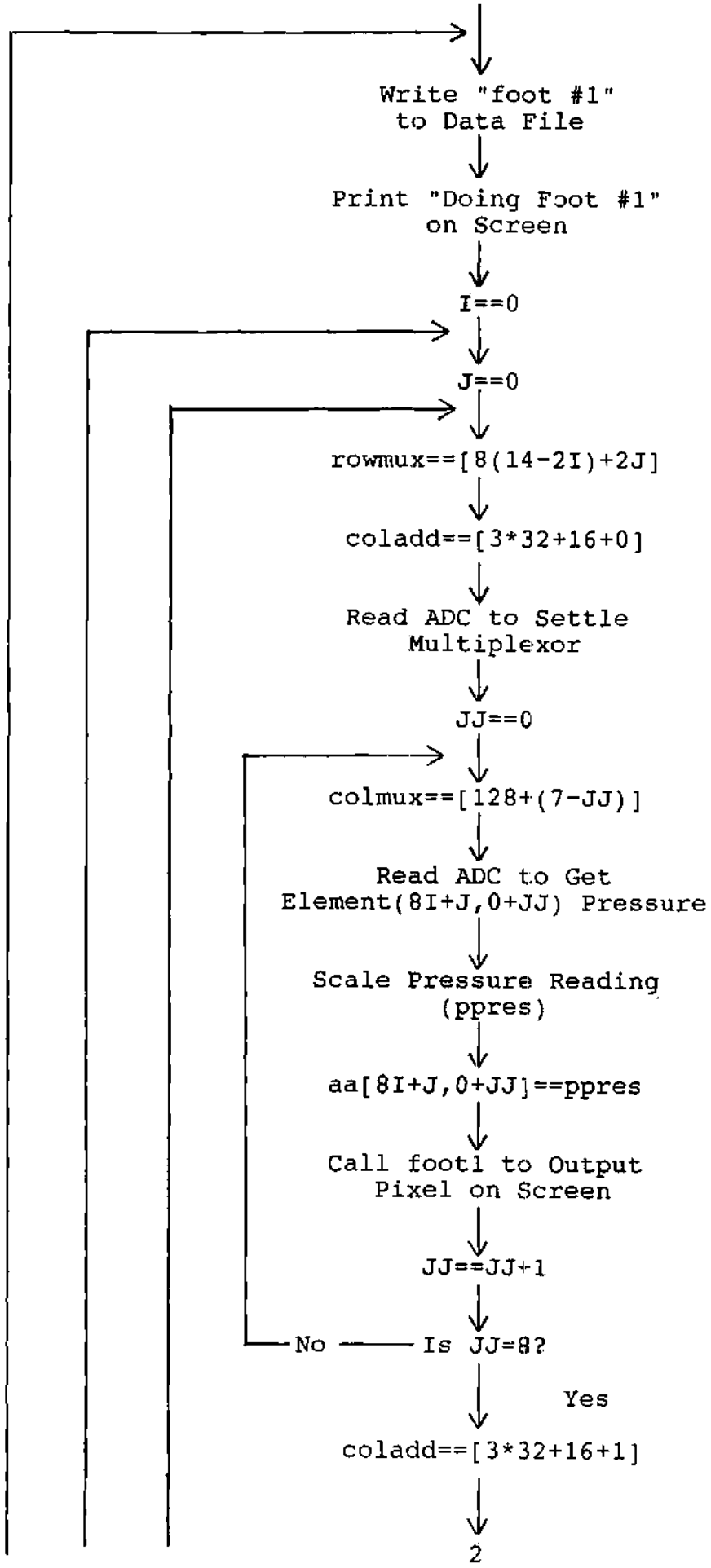
```
    asm     mov     dx,byt6
    asm     out     dx,al
    asm     mov     dx,byt7
    asm     out     dx,al
    asm     mov     dx,byt8
    asm     out     dx,al
    asm     pop     dx
    asm     pop     ax
    }

int inpres()
/* This function inputs a pressure pixel from the DT-2814 adc board */
    {
    asm     push    ax
    asm     push    dx
    asm     push    cx

    asm     mov     ax,rowmux
    asm     mov     dx,byt0
    asm     out     dx,al   /* output the row mux addr */

    asm     mov     ax,colmux
    asm     mov     dx,byt2
    asm     out     dx,al   /* output the column mux addr */

    asm     mov     cx,50   /* wait 50 micro seconds to let settle*/
    inpre1:
    asm     cmp     cx,0
    asm     dec     cx
    asm     jne     inpre1

    asm     mov     ax,colad
    asm     mov     dx,adcr
    asm     out     dx,al   /* output adc control to start conversion */

    asm     mov     cx,25   /* wait 25 micro seconds */
    inpre2:
    asm     cmp     cx,0
    asm     dec     cx
    asm     jne     inpre2

    inpre3:
    asm     mov     dx,adsr
    asm     in      al,dx
    asm     and     al,0x80
    asm     cmp     al,0
    asm     je      inpre3      /* if bit 7 not set is busy check again*

    asm     mov     dx,addata
    asm     in      al,dx
    asm     mov     ah,al
    asm     in      al,dx   /* read in the high and low adc data bytes *
    asm     mov     upres,ax

    asm     pop     cx
```

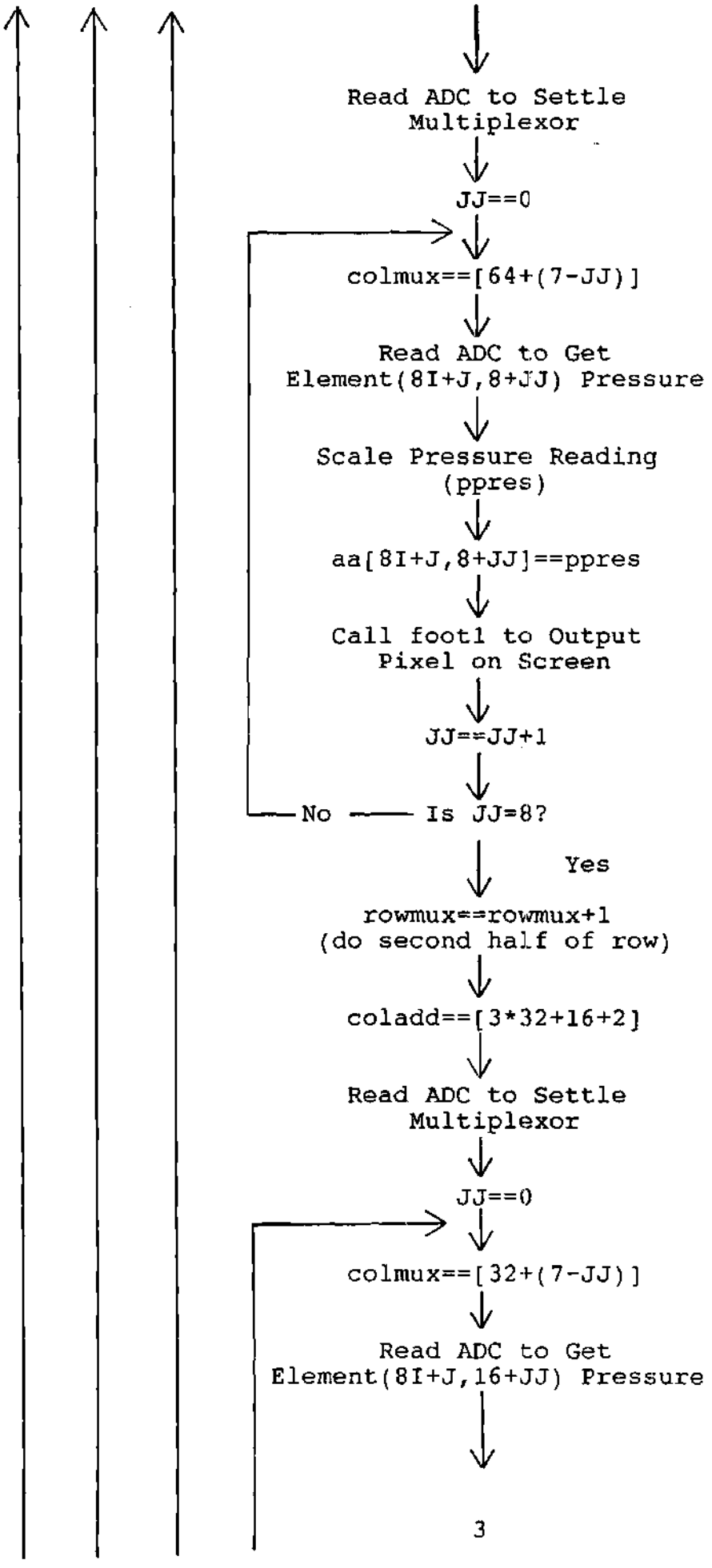
```
   asm     pop     dx
   asm     pop     ax
   }

int foot1(int x,int y,int pres)
/* This function plots a pressure pixel for foot 1 */
   {
   int i,ix,iy;

   ix=5*x+220;
   iy=4*(63-y)+20;
   setcolor(pres);
   for(i=0;i<4;i++)
      {
      line(ix,349-iy+i,ix+5,349-iy+i);
      }
   }

int foot2(int x,int y,int pres)
/* This function plots a pressure pixel for foot 2 */
   {
   int i,ix,iy;

   ix=5*x+400;
   iy=4*(63-y)+20;
   setcolor(pres);
   for(i=0;i<4;i++)
      {
      line(ix,349-iy+i,ix+4,349-iy+i);
      }
   }

int colorkey(int x,int y)
/* This function plots the pressure color key at x,y */
   {
   int i,j,ix,iy;

   ix=x;
   for(j=0;j<16;j++)
      {
      iy=y+12*j;
      setcolor(j);
      for(i=0;i<12;i++)
         {
         line(ix,349-iy+i,ix+14,349-iy+i);
         }
      }
   setcolor(WHITE);
   outtextxy(x+18,349-2+4-y," 0");
   outtextxy(x+18,349-14+4-y," 1");
   outtextxy(x+18,349-26+4-y," 2");
   outtextxy(x+18,349-38+4-y," 3");
   outtextxy(x+18,349-50+4-y," 4");
   outtextxy(x+18,349-62+4-y," 5");
   outtextxy(x+18,349-74+4-y," 6");
```

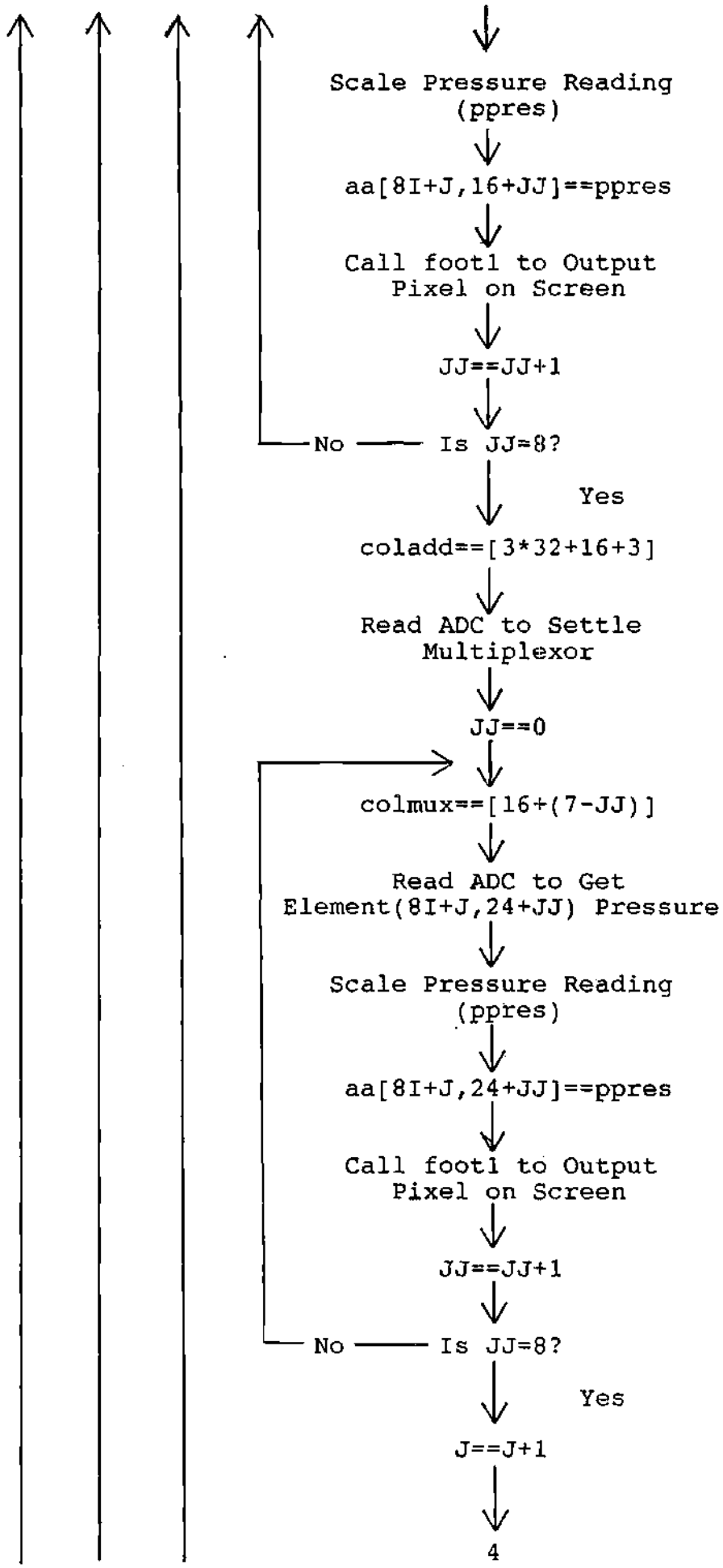
```
outtextxy(x+18,349-86+4-y," 7");
outtextxy(x+18,349-98+4-y," 8");
outtextxy(x+18,349-110+4-y," 9");
outtextxy(x+18,349-122+4-y,"10");
outtextxy(x+18,349-134+4-y,"11");
outtextxy(x+18,349-146+4-y,"12");
outtextxy(x+18,349-158+4-y,"13");
outtextxy(x+18,349-170+4-y,"14");
outtextxy(x+18,349-182+4-y,"15");
}
```

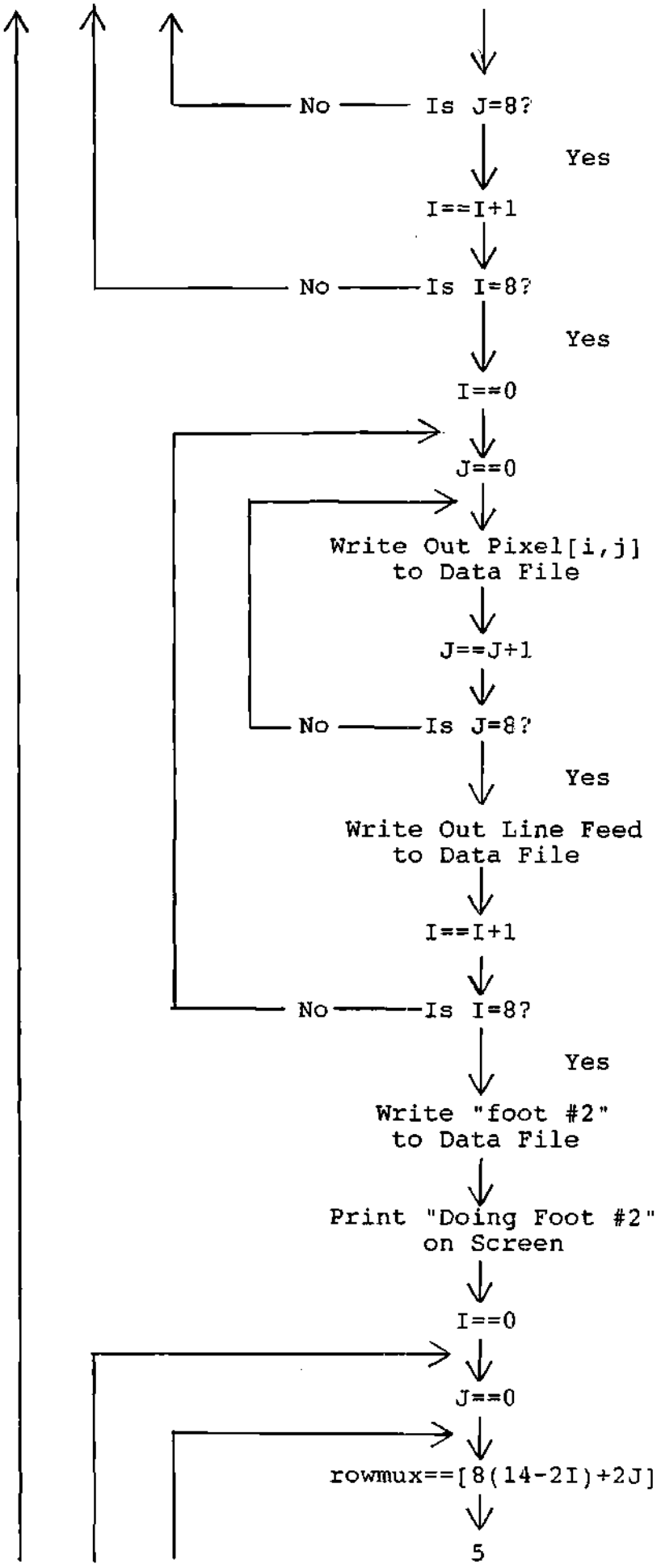
FLOWCHART

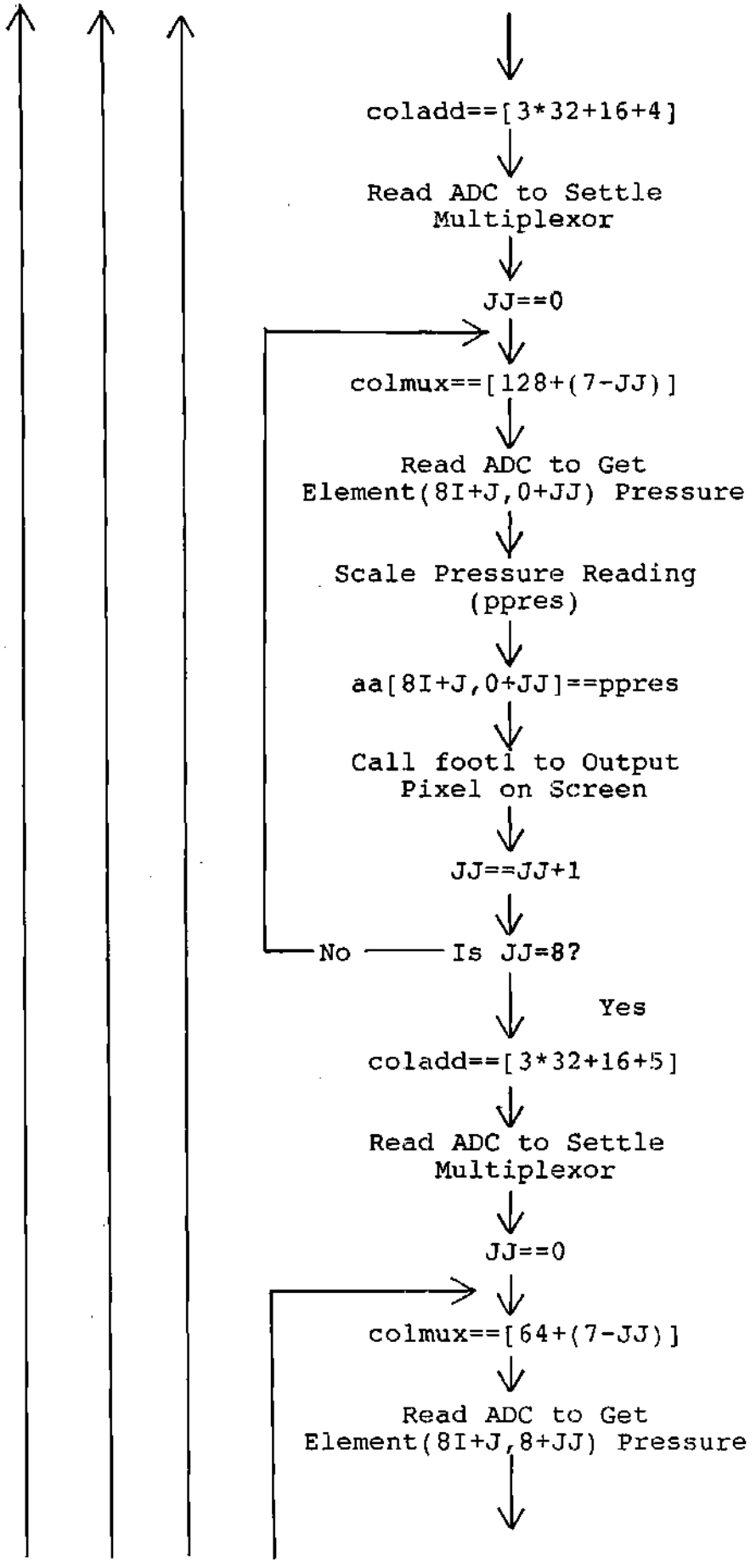

Initialize Variables for Interface Boards
Reset Qua Tech PXB-721 Parallel I/O Board
Zero All Qua Tech Outputs
Reset DT-2814 ADC Board
Get Name for Output Data File
Create a Write Only File for Pressure Data
Get Number of Cycles (nrl)
Write nrl to Data File
Has Return Key Been Pressed?
No
Yes
Put Video Display in Graphics Mode
Write Text Headings and Color Key to Screen
K==0

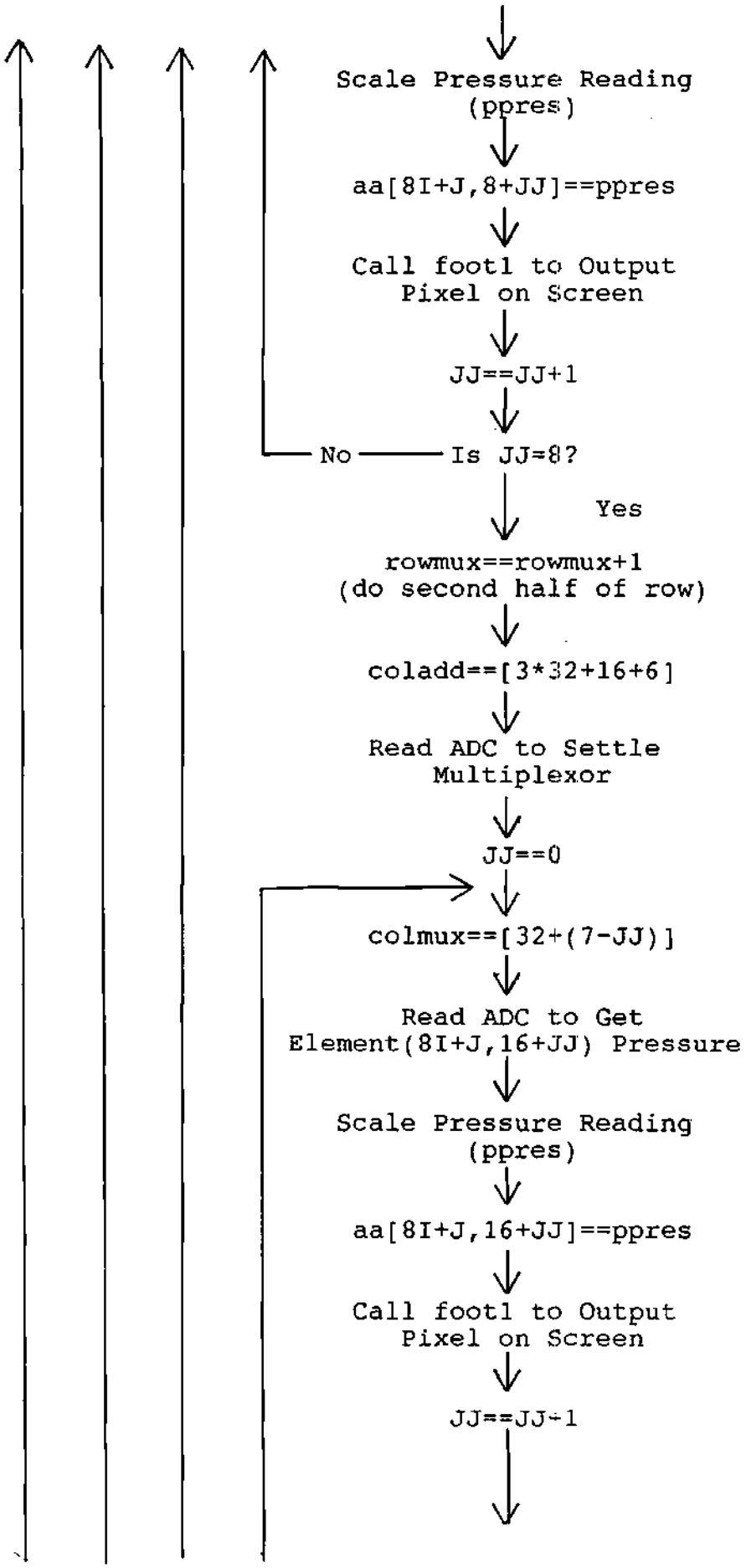

Write "foot #1"
to Data File
Print "Doing Foot #1"
on Screen
I==0
J==0
rowmux==[8(14-2I)+2J]
coladd==[3*32+16+0]
Read ADC to Settle
Multiplexor
JJ==0
colmux==[128+(7-JJ)]
Read ADC to Get
Element(8I+J,0+JJ) Pressure
Scale Pressure Reading
(ppres)
aa[8I+J,0+JJ]==ppres
Call foot1 to Output
Pixel on Screen
JJ==JJ+1
No
Is JJ=8?
Yes
coladd==[3*32+16+1]
2

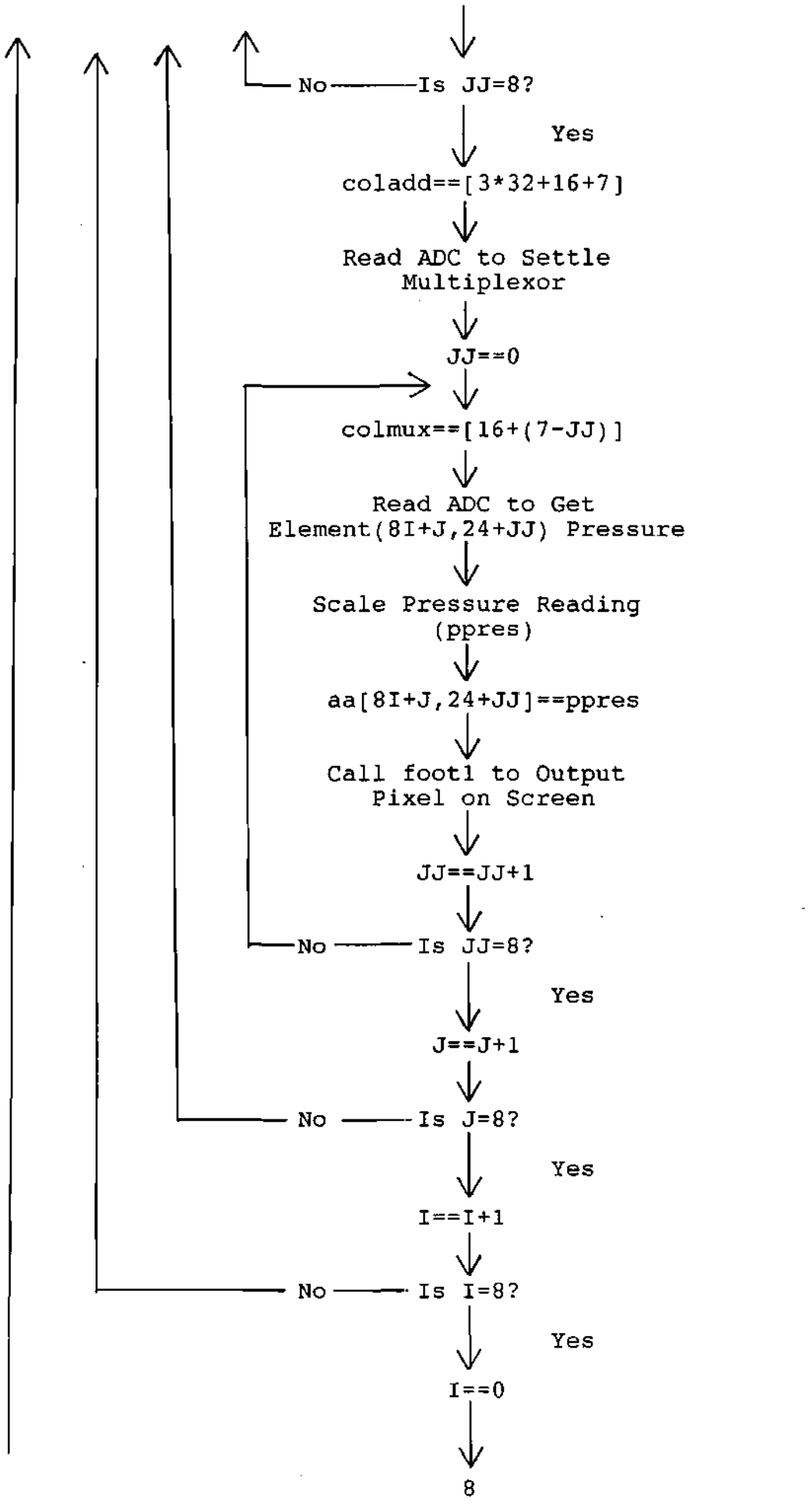

Read ADC to Settle Multiplexor
JJ==0
colmux==[64+(7-JJ)]
Read ADC to Get Element(8I+J,8+JJ) Pressure
Scale Pressure Reading (ppres)
aa[8I+J,8+JJ]==ppres
Call foot1 to Output Pixel on Screen
JJ==JJ+1
Is JJ=8?
No
Yes
rowmux==rowmux+1 (do second half of row)
coladd==[3*32+16+2]
Read ADC to Settle Multiplexor
JJ==0
colmux==[32+(7-JJ)]
Read ADC to Get Element(8I+J,16+JJ) Pressure

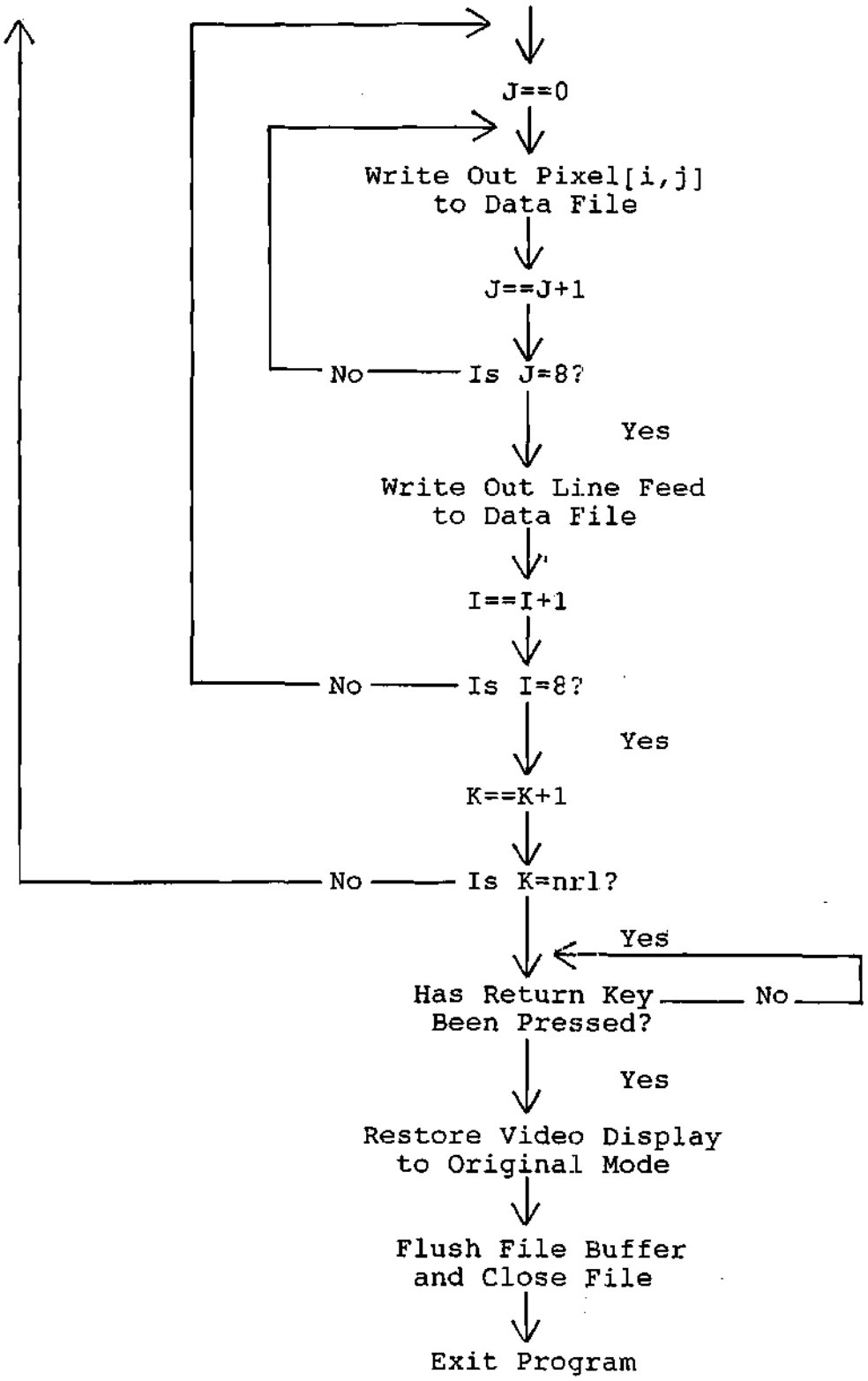

Scale Pressure Reading (ppres)
aa[8I+J,16+JJ]==ppres
Call foot1 to Output Pixel on Screen
JJ==JJ+1
No
Is JJ=8?
Yes
coladd==[3*32+16+3]
Read ADC to Settle Multiplexor
JJ==0
colmux==[16+(7-JJ)]
Read ADC to Get Element(8I+J,24+JJ) Pressure
Scale Pressure Reading (ppres)
aa[8I+J,24+JJ]==ppres
Call foot1 to Output Pixel on Screen
JJ==JJ+1
No
Is JJ=8?
Yes
J==J+1
4


No
Is J=8?
Yes
I==I+1
No
Is I=8?
Yes
I==0
J==0
Write Out Pixel[i,j]
to Data File
J==J+1
No
Is J=8?
Yes
Write Out Line Feed
to Data File
I==I+1
No
Is I=8?
Yes
Write "foot #2"
to Data File
Print "Doing Foot #2"
on Screen
I==0
J==0
rowmux==[8(14-2I)+2J]
5


coladd==[3*32+16+4]
Read ADC to Settle Multiplexor
JJ==0
colmux==[128+(7-JJ)]
Read ADC to Get Element(8I+J,0+JJ) Pressure
Scale Pressure Reading (ppres)
aa[8I+J,0+JJ]==ppres
Call foot1 to Output Pixel on Screen
JJ==JJ+1
No
Is JJ=8?
Yes
coladd==[3*32+16+5]
Read ADC to Settle Multiplexor
JJ==0
colmux==[64+(7-JJ)]
Read ADC to Get Element(8I+J,8+JJ) Pressure


Scale Pressure Reading
(ppres)
aa[8I+J,8+JJ]==ppres
Call foot1 to Output
Pixel on Screen
JJ==JJ+1
No
Is JJ=8?
Yes
rowmux==rowmux+1
(do second half of row)
coladd==[3*32+16+6]
Read ADC to Settle
Multiplexor
JJ==0
colmux==[32+(7-JJ)]
Read ADC to Get
Element(8I+J,16+JJ) Pressure
Scale Pressure Reading
(ppres)
aa[8I+J,16+JJ]==ppres
Call foot1 to Output
Pixel on Screen
JJ==JJ+1


No
Is JJ=8?
Yes
coladd==[3*32+16+7]
Read ADC to Settle Multiplexor
JJ==0
colmux==[16+(7-JJ)]
Read ADC to Get Element(8I+J,24+JJ) Pressure
Scale Pressure Reading (ppres)
aa[8I+J,24+JJ]==ppres
Call foot1 to Output Pixel on Screen
JJ==JJ+1
No
Is JJ=8?
Yes
J==J+1
No
Is J=8?
Yes
I==I+1
No
Is I=8?
Yes
I==0
8


J==0
Write Out Pixel[i,j]
to Data File
J==J+1
No
Is J=8?
Yes
Write Out Line Feed
to Data File
I==I+1
No
Is I=8?
Yes
K==K+1
No
Is K=nrl?
Yes
Has Return Key
Been Pressed?
No
Yes
Restore Video Display
to Original Mode
Flush File Buffer
and Close File
Exit Program

APPENDIX D

System Operation Instructions

INSTRUCTIONS FOR OPERATION OF THE PROGRAM

1. Turn on the computer and wait for DOS prompt.

2. Turn on power to sensing instrument.

3. Type "CD FEET" to change to directory FEET.

4. Run program by typing "FEET7".

5. Type file name to store pressure data and press RETURN.

6. Type the number of cycles for sensing pressure and collecting data (1-20), then press RETURN.

7. The patient will stand on the two pads (left and right), then press RETURN.

8. The program will commence taking data, and the colored image of pressure points and cycle number will appear on the screen. Also, the legend will appear which is represented with brighter colors correlated with increasing pressure.

9. When the data collection is finished, press RETURN to return to DOS.

10. The pressure sensing data (numeric pressure values) can be viewed with any general purpose editor. A viewing program for "ascii" files has been included in the program. To view the data (numeric pressure values per pixel), type "VIEW file name and extension" (eg., BROWN.1). Use the cursor keys to move in a direction within the file data. Press ESC to exit the VIEW program.

11. Print the data using a word processing program (eg., WordStar or WordPerfect) in non-document mode.